CURSO PRÁTICO 35 DE PROGRAMAÇÃO DE COMPUTADORES

INPUT

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

NOVA CULTURAL

Cz$ 75,00

INPUT

Vol. 3 Nº 35

NESTE NÚMERO

PROGRAMAÇÃO DE JOGOS

SERRA PELADA: O TOQUE DE MIDAS

O rei Midas tinha o dom de transformar em ouro tudo o que tocava. Você, em contrapartida, terá que fazer levantamentos geológicos, calcular custos e realizar escavações. Mas não se desespere: o computador facilitará o seu trabalho, exibindo gráficos ilustrativos 681

PROGRAMAÇÃO BASIC

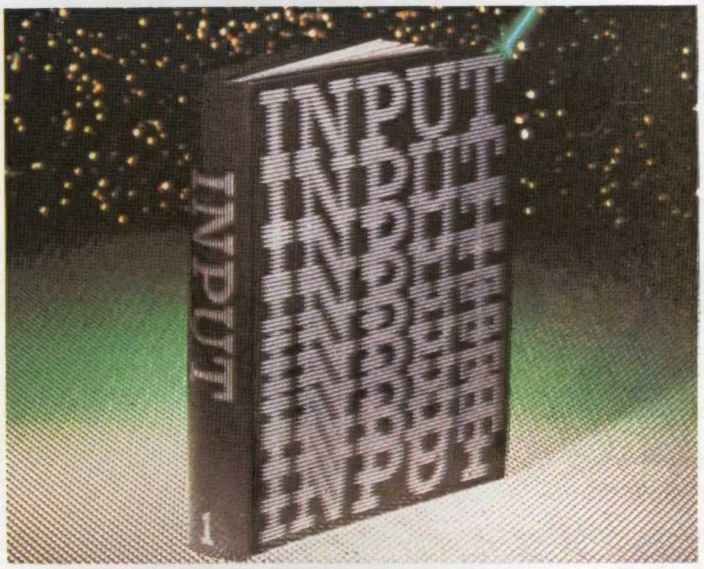
RECORRA AOS ARQUIVOS

Programas e arquivos de dados. Limitações da fita cassete. Transferência de dados de um programa para outro. O que é um arquivo de dados? Significados de ler (*read*) e escrever (*write*) para um computador 688

PROGRAMAÇÃO BASIC

PROGRAMAÇÃO DE GRÁFICOS EM 3-D (4)

Agora que você já conhece os segredos da perspectiva, está em condições de reunir vários programas e rotinas e desenhar as mais diversas figuras flutuando nas profundezas do espaço interestelar..... 694



PLANO DA OBRA
INPUT é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

FÉRIAS, VIAGENS, MUDANÇAS...
NÃO FIQUE COM A COLEÇÃO INCOMPLETA
Se você está saindo de férias, pretende viajar ou vai se ausentar por algum tempo, avise antecipadamente seu jornaleiro. Ele pode guardar os seus fascículos enquanto você estiver fora. Se, por qualquer motivo, você perdeu alguns números, peça-os também a seu jornaleiro, ou entre em contato com nossa Distribuidora:

1. **Pessoalmente** — Em *São Paulo,* os endereços são: Rua Brigadeiro Tobias, 773, Centro, telefone 227-4188; Av. Industrial, 117, Santo André, telefone 449-0411, das 7h30 às 17h00 - dias úteis. No *Rio de Janeiro,* Av. Mem de Sá, 191/193, Centro, telefone (021) 222-7422, das 7h30 às 17h00 - dias úteis.
2. **Por carta** — Envie para:
   DINAP — Distribuidora Nacional de Publicações
   Números Atrasados
   Estrada Velha de Osasco, 132 — Jardim Teresa
   CEP 06040 — Osasco — SP
3. **Por telex** — Utilize o nº (011) 33 670 DNAP.

Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações Lda. — Qta. Pau Varais, Azinhaga de Fetais, 2685, Camarate, Lisboa; Apartado 57; Télex 43 069 JARLIS P.
Atenção: Após seis meses do encerramento da coleção, o atendimento dos pedidos dependerá da disponibilidade do estoque.
Obs.: Quando pedir livros, mencione sempre o título e/ou autor da obra, além do número da edição.

COLABORE CONOSCO
Encaminhe seus comentários, críticas, sugestões ou reclamações ao
SERVIÇO DE ATENDIMENTO AO LEITOR
Caixa Postal 9 442, São Paulo — SP.

EDITOR
RICHARD CIVITA
NOVA CULTURAL

Presidente
Flávio Barros Pinto
Diretoria
Carmo Chagas, Iara Rodrigues,
Pierluigi Bracco, Plácido Nicoletto,
Roberto Silveira, Shozi Ikeda,
Sônia Carvalho

REDAÇÃO
Diretor Editorial: Carmo Chagas
Editores Executivos:
Berta Sztark Amar, Stefania Crema
Editor Chefe: Paulo de Almeida
Editoras Assistentes: Ana Lúcia B. de Lucena,
Marisa Soares de Andrade
Chefe de Arte: Carlos Luiz Batista
Assistentes de Arte: Dagmar Bastos Sampaio,
Grace Alonso Arruda, Monica Lenardon Corradi

Secretário de Redação: Mauro de Queiroz

Colaboradores
Consultor Editorial Responsável:
Dr. Renato M. E. Sabbatini
(Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas-SP)
Execução Editorial: DATAQUEST Assessoria em Informática Ltda., Campinas
Tradução, adaptação, programação e redação:
Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Marcos Huascar Velasco, Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
Coordenação Geral: Rejane Felizatti Sabbatini

COMERCIAL
Diretor Comercial: Roberto Silveira
Gerente Comercial: Joaquim Celestino da Silva
Gerente de Circulação: Denise Mozol
Gerente de Propaganda e Publicidade: José Carlos Madio
Gerente de Pesquisa e Análise de Mercado:
Wagner M. P. Nabuco de Araújo

CLC

A Editora Nova Cultural Ltda. é uma empresa do Grupo CLC — Comunicações, Lazer e Cultura S.A.
Presidente: Richard Civita
Diretoria: Flávio Barros Pinto, João Gomez, Menahem M. Politi, Renê C.X. Santos, Stélio Alves Campos

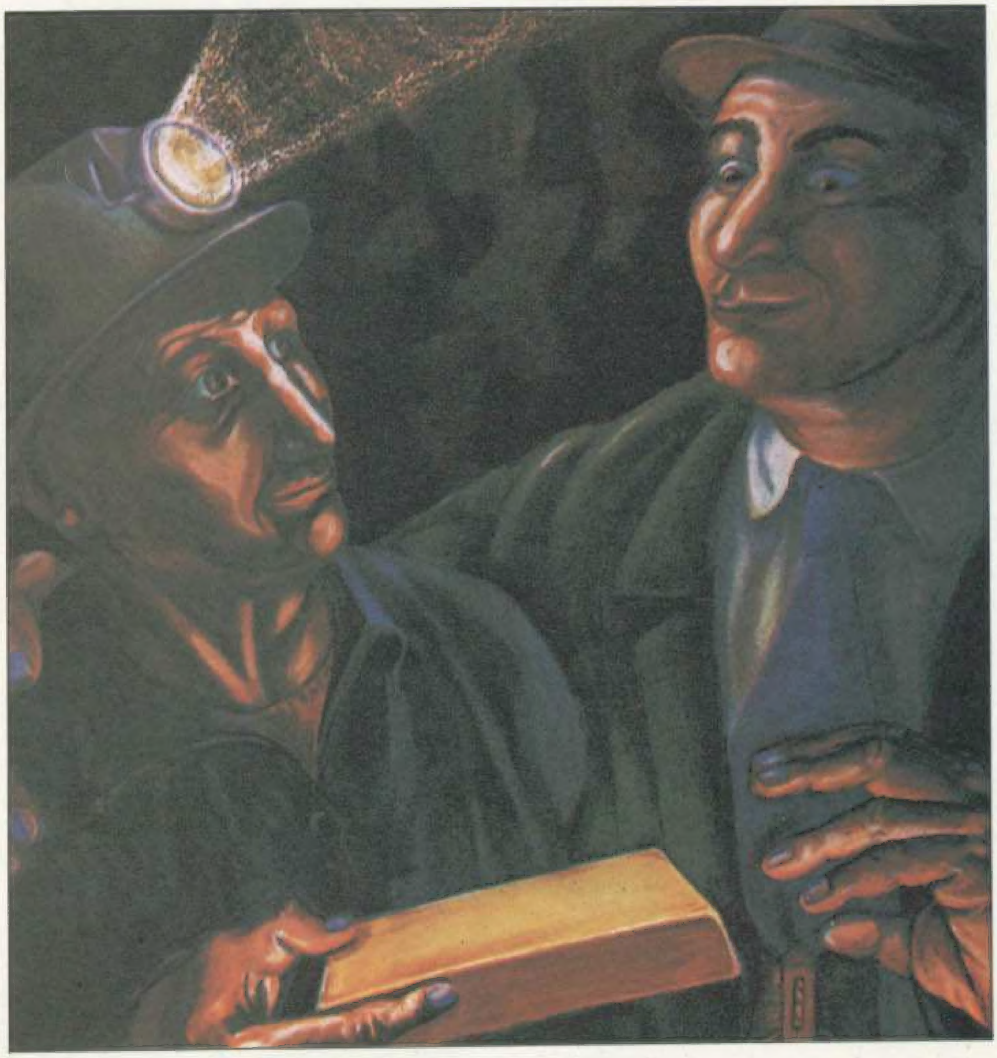
© Marshall Cavendish Limited, 1984/85.
© Editora Nova Cultural Ltda., São Paulo, Brasil, 1986; 2ª edição, 1987.

Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, 2000 - 3º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta pela AM Produções Gráficas Ltda. e impressa pela Companhia Lithographica Ypiranga.

SERRA PELADA: O TOQUE DE MIDAS

- ADICIONE AS ROTINAS QUE FALTAVAM
- LEVANTAMENTO GEOLÓGICO
- ESCAVAÇÕES
- GRÁFICOS ILUSTRATIVOS

Figura da mitologia grega, o rei Midas recebeu de Dioniso (deus do vinho) o poder de transformar em ouro tudo o que tocava. Em nosso jogo, porém, as coisas não serão assim tão fáceis.



A primeira parte deste jogo apresentou opções como "Pesquisa e Desenvolvimento", "Levantamento Geológico", "Cavar mais 200 m", "Vender Ouro no Mercado" e "Passar a Vez". As subrotinas que se seguem completam o programa, particularizando cada opção.

A opção "Pesquisa e Desenvolvimento" é obtida por meio da tecla 1; "Levantamento Geológico" é a opção 2; "Cavar mais 200 m", a opção 3; "Vender Ouro no Mercado", a opção 4. A quinta opção cuida de "Passar a Vez", de forma que nenhuma rotina especial é necessária. As opções 1, 2 e 4 introduzem um elemento de acaso em seu resultado, tentando, assim, imitar a realidade do trabalho de mineração.

PESQUISA E DESENVOLVIMENTO

```
1000 BORDER 6: PAPER 6: INK 0:
CLS
1010 PRINT  PAPER 1; INK 6;AT 3
,4;" PESQUISA E DESENVOLVIMENTO
 ";AT 4,6;"(para reduzir os cus
tos)"
1020 PRINT AT 7,4;"Quanto prete
nde investir?($)": INPUT rd
1050 LET a(m,4)=a(m,4)-INT (rd*
.05)-1
1060 IF a(m,4)<0 THEN  LET a(m,
4)=0
1080 LET a(m,2)=a(m,2)-rd: LET
a(m,1)=a(m,1)-rd
1100 PRINT AT 13,3;"Os custos d
e mineracao foram";TAB 3;"reduz
idos em $";INT (rd*.05)+1;" por
 200 m"
1110 FOR z=1 TO 300: NEXT z
1120 RETURN
```

```
1000 GOSUB 1500
1010 LOCATE 1,0:PRINT "Pesquisa
 e Desenvolvimento":LOCATE 1,1:
PRINT " (para reduzir os custos
) "
1020 LOCATE 0,3:PRINT "Quanto p
retende":INPUT "investir ";RD
1030 IF RD<=0 THEN 1000
1050 A(M,3)=A(M,3)-INT(RD/20)-1
1060 IF A(M,3)<0 THEN A(M,3)=0
1080 A(M,1)=A(M,1)-RD:A(M,0)=A(
M,0)-RD
1100 PRINT:PRINT "Os custos de
mineração foram":PRINT "reduzid
os em $";INT(RD/20)+1;"por 200
m"
1110 FOR Z=1 TO 2000:NEXT
1120 RETURN
```

```
1000  HOME
1010  VTAB 1: HTAB 5: INVERSE :
 PRINT " PESQUISA E DESENVOLVIM
ENTO ": NORMAL : PRINT : PRINT
 TAB( 10)"P/REDUZIR OS CUSTOS"
1020  PRINT : PRINT : INPUT "QU
ANTO PRETENDE INVESTIR ";RD
1030  IF RD < 0 THEN 1000
1050 A(M,3) = A(M,3) -  INT (RD
 / 20) - 1
1060  IF A(M,3) < 0 THEN A(M,3)
 = 0
1080 A(M,1) = A(M,1) - RD:A(M,0
) = A(M,0) - RD
1100  PRINT : PRINT "OS CUSTOS
DE MINERACAO FORAM REDUZIDOS":
PRINT "EM $"; INT (RD / 20) + 1
;" POR 200 M"
1110  FOR Z = 1 TO 2000: NEXT
1120  RETURN
```

```
1000 CLS
1010 PRINT @2,"pesquisa e desen
volvimento":PRINT @34,"(PARA RE
DUZIR OS CUSTOS)"
1020 PRINT:INPUT"QUANTO PRETEND
E INVESTIR ";RD
1030 IF RD<0 THEN 1000
1050 A(M,3)=A(M,3)-INT(RD/20)-1
1060 IF A(M,3)<0 THEN A(M,3)=0
1080 A(M,1)=A(M,1)-RD:A(M,0)=A(
M,0)-RD
1100 PRINT @257,"OS CUSTOS DE M
INERACAO FORAM    REDUZIDOS EM
";INT(RD/20)+1;"POR 200 M"
1110 FOR Z=1 TO 2000:NEXT
1120 RETURN
```

No programa do Spectrum, a linha 1000, além de limpar a tela, muda suas cores. Nos demais computadores, as cores da tela permanecem as mesmas após a limpeza do vídeo.

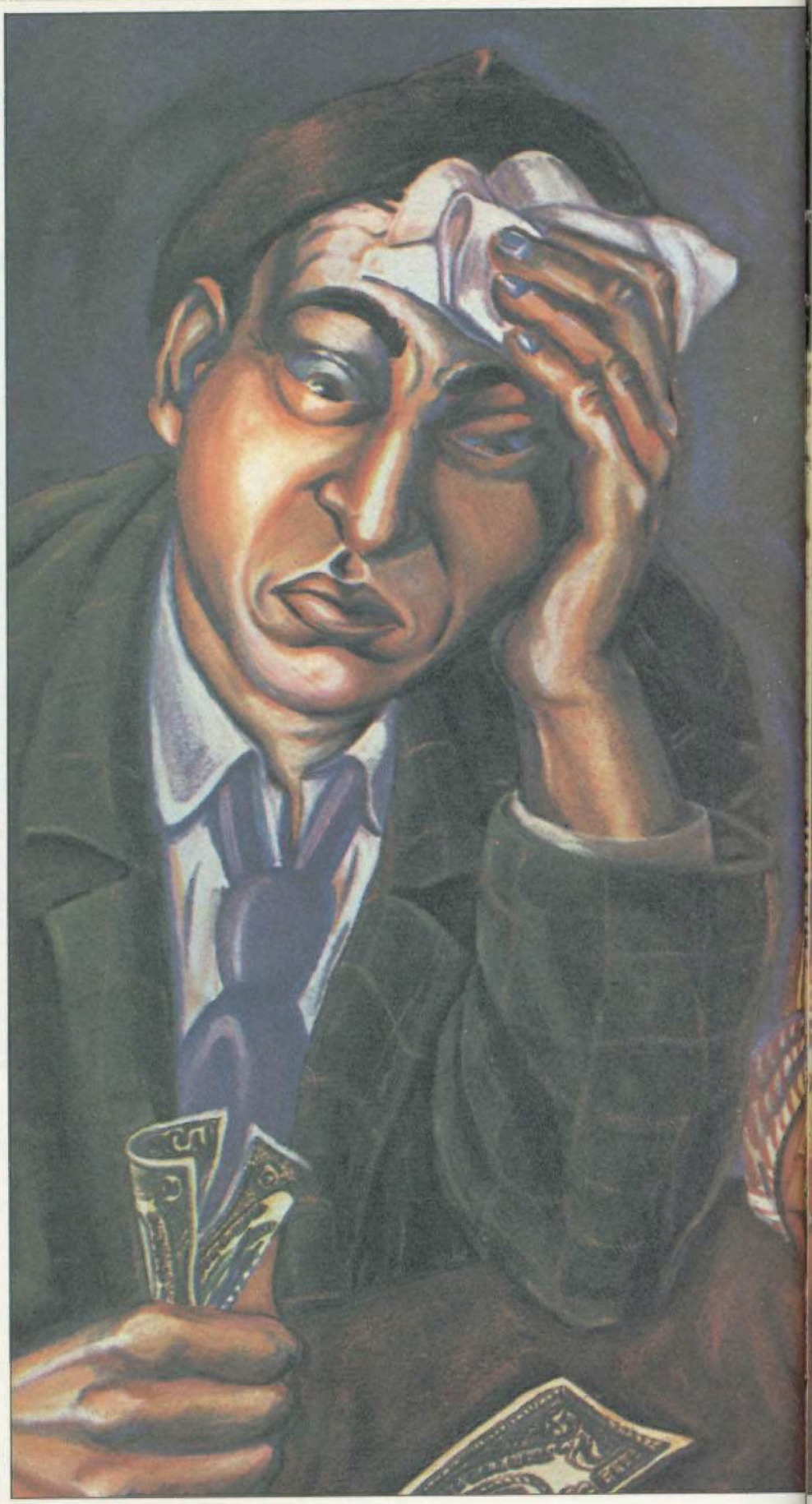
A linha 1010 coloca um rótulo no alto da tela, antes que a linha 1020 pergunte ao jogador quanto dinheiro será investido em pesquisa e desenvolvimento (**RD**, ou **rd** no Spectrum, contém a quantia escolhida).

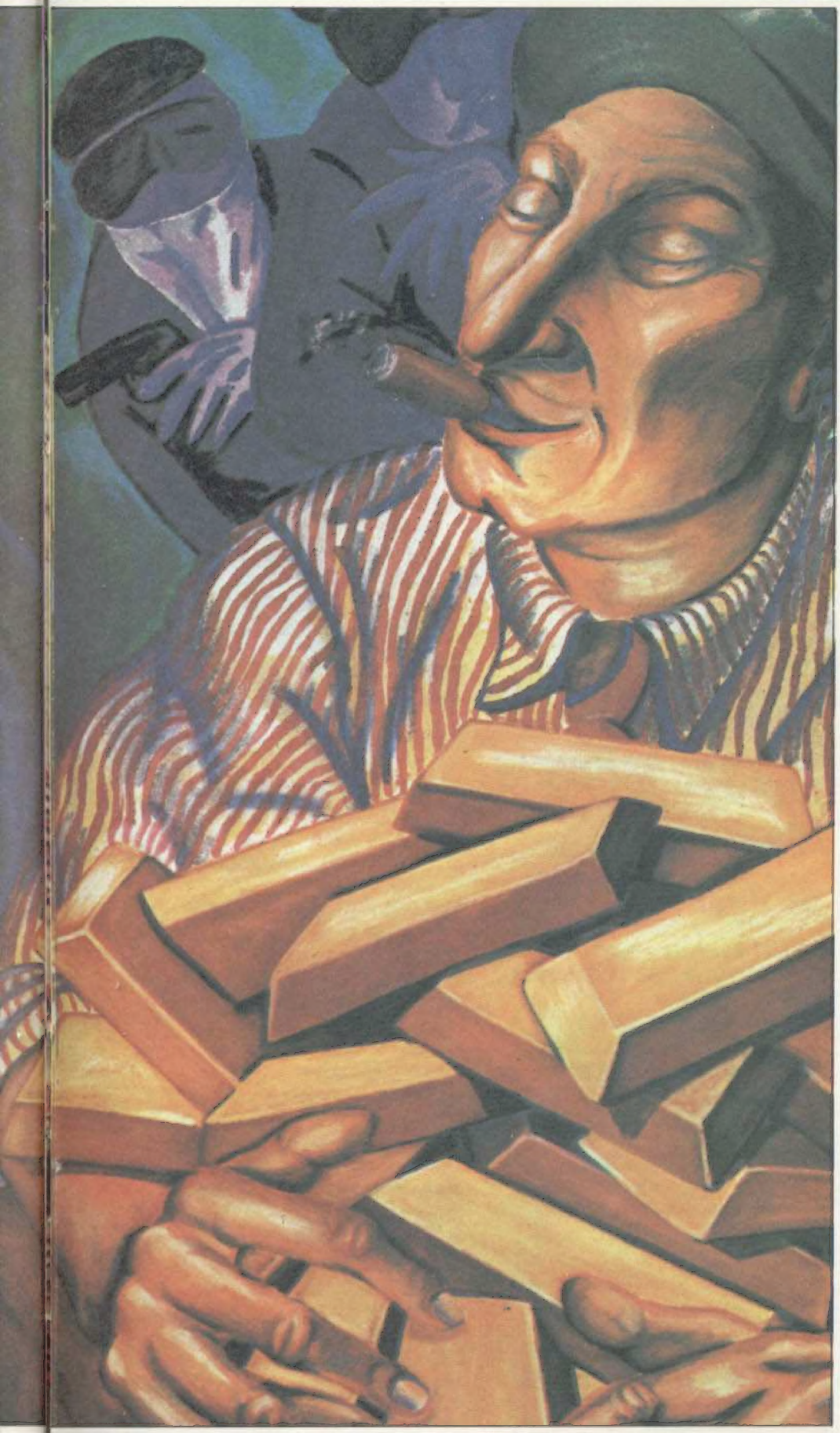
A linha 1050 diminui o custo de mineração de acordo com a quantia investida no processo de pesquisa. A linha 1060 assegura que o custo de produção não se torne negativo. A linha 1080 acerta o saldo total e a quantia do jogador, de acordo com o que foi gasto em "Pesquisa e Desenvolvimento".

A linha 1100 informa qual foi a redução de custos por 200 m de escavação. A linha 1110 contém um laço **FOR...NEXT** que provoca uma pequena demora antes que a sub-rotina termine.

LEVANTAMENTO GEOLÓGICO

```
2000 PAPER 4: BORDER 4: INK 0:
CLS
2030 LET r(m)=0: LET c(m,1)=INT
 (RND*90)+10: LET c(m,2)=INT ((
RND*5)+2)*200: LET c(m,3)=INT (
RND*200)+1: LET 11=INT (RND*3)-
1
2050 LET c(m,4)=c(m,2)+11*200
2070 LET c(m,5)=0: LET kk=INT (
RND*100): IF kk<c(m,1) THEN  LE
T c(m,5)=1
2080 PRINT  PAPER 6; INK 0;AT 2
,5;" LEVANTAMENTO GEOLOGICO ":
PRINT AT 5,5;"Probabilidade de"
: PRINT AT 6,5;"encontrar ouro:
        ";c(m,1);"%": PRINT AT
8,5;"Provavel profundidade: ";c
(m,2);"m": PRINT AT 10,5;"Prova
vel quantidade: ";c(m,3);"kg"
2100 LET z=INT (RND*150000): LE
```





```
T a(m,2)=a(m,2)-z: LET a(m,1)=a
(m,1)-z
2110 PRINT  FLASH 1;AT 12,0;"De
seja  comecar a escavar (s/n)?"
2120 LET r$=INKEY$: IF r$="" TH
EN  GOTO 2120
2130 IF r$="s" THEN  LET a(m,6)
=0: LET r(m)=1: GOTO 3000
2500 RETURN
```

```
2000 GOSUB 1500
2030 R(M)=0:C(M,0)=INT(RND(1)*9
0)+10:C(M,1)=INT(RND(1)*5+2)*20
0:C(M,2)=INT(RND(1)*200+1):LL=I
NT(RND(1)*3)-1
2050 C(M,3)=C(M,1)+LL*200
2070 C(M,4)=0:KK=INT(RND(1)*100
):IF KK<C(M,0) THEN C(M,4)=1
2080 LOCATE 3,0:PRINT "Levantam
ento Geológico":PRINT:PRINT "Pr
obabilidade de":PRINT "encontra
r ouro:";C(M,0);"%"
2090 PRINT:PRINT "Provável":PRI
NT "profundidade:";C(M,1);"m":P
RINT:PRINT "Provável":PRINT "qu
antidade:";C(M,2);"kg"
2100 Z=INT(150000!*RND(1)):A(M,
1)=A(M,1)-Z:A(M,0)=A(M,0)-Z
2110 PRINT:PRINT "Deseja começa
r as":PRINT "escavações (S/N) ?
"
2120 R$=INKEY$:IF R$<>"S" AND R
$<>"N" THEN 2120
2130 IF R$="S" THEN A(M,5)=0:R(
M)=1:GOTO 3000
2500 RETURN
```

```
2000  HOME
2030 R(M) = 0:C(M,0) =  INT ( R
ND (1) * 90) + 10:C(M,1) = 200
*  INT ( RND (1) * 5 + 1):C(M,2
) =  INT ( RND (1) * 200 + 1):L
L =  INT ( RND (1) * 3) - 1
2050 C(M,3) = C(M,1) + LL * 200
2070 C(M,4) = 0:KK =  INT ( RND
 (1) * 100): IF KK < C(M,0) THE
N C(M,4) = 1
2080  VTAB 1: HTAB 7: INVERSE :
 PRINT " LEVANTAMENTO GEOLOGICO
 ": NORMAL : PRINT : PRINT : PR
INT "PROBABILIDADE DE": PRINT "
ENCONTRAR OURO: ";C(M,0);"%"
2090  PRINT : PRINT "PROVAVEL P
ROFUNDIDADE: ";C(M,1);" M": PRI
NT : PRINT "PROVAVEL QUANTIDADE
: ";C(M,2);" KG"
2100 Z =  INT ( RND (1) * 15000
0):A(M,1) = A(M,1) - Z:A(M,0) =
 A(M,0) - Z
2110  PRINT : PRINT "DESEJA COM
ECAR AS ESCAVACOES (S/N) ?"
2120  GET R$
2130  IF R$ = "S" THEN A(M,5) =
 0:R(M) = 1: GOTO 3000
2500  RETURN
```

```
2000 CLS
```

```
2030 R(M)=0:C(M,0)=RND(90)+9:C(
M,1)=(RND(5)+1)*200:C(M,2)=RND(
200):LL=RND(3)-2
2050 C(M,3)=C(M,1)+LL*200
2070 C(M,4)=0:KK=RND(100)-1:IF
KK<C(M,0) THEN C(M,4)=1
2080 PRINT @5,"LEVANTAMENTO GEO
LOGICO ":PRINT @129,"PROBABIL.D
E ENCONTRAR OURO";C(M,0);"%":PR
INT @193,"PROVAVEL PROFUNDIDADE
 ";C(M,1);"M":PRINT @257,"PROVA
VEL QUANTIDADE ";C(M,2);"KG"
2100 Z=RND(150000)-1:A(M,1)=A(M
,1)-Z:A(M,0)=A(M,0)-Z
2110 PRINT @353,"DESEJA COMECAR
 AS ESCAVACOES ? "
2120 R$=INKEY$:IF R$<>"S" AND R
$<>"N" THEN 2120
2130 IF R$="S" THEN A(M,5)=0:R(
M)=1:GOTO 3000
2500 RETURN
```

Todas as máquinas limpam a tela na linha 2000. O Spectrum também muda as cores da tela. A linha 2030 faz com que o valor de **R(M)** — **r(m)**, no caso do Spectrum — seja zero, para indicar que as escavações ainda não começaram. Essa linha também calcula as possibilidades de encontrar ouro, além da quantidade de metal e da profundidade provável do veio. **LL** (ou **ll**) é um número aleatório entre –1 e 1 que será usado na próxima linha para determinar a profundidade real da mina (lembre-se de que o valor em **C(M,2)** é apenas a profundidade provável).

A linha 2050 faz com que **C(M,4)** assuma um valor igual a **C (M,2)** (mais ou menos 200 m, ou seja, 200 vezes **LL**). A seguir, a linha 2070 decide se a mina vai ou não ter ouro. Se não houver nenhum ouro, **C(M,5)** se tornará zero. **KK** é outro número aleatório, entre 0 e 99. Ele é comparado com as chances de se encontrar ouro — se for menor do que estas, **C(M,5)** se tornará 1 para indicar que há ouro na mina.

A linha 2080 (2090 no Apple, no TK-2000 e no MSX) dá ao proprietário o resultado do levantamento geológico. Os números informam as chances de se encontrar ouro, onde e em que quantidade. Mas a existência ou não do metal na mina depende de uma série de fatores aleatórios. Dessa forma, o jogador deve ser muito criterioso ao fazer seus investimentos.

Agora, as más notícias: o relatório é pago e seu custo, imprevisível. Em todo caso, a conta fica no máximo em $150000 (valor de Z escolhido na linha 2100). O custo do relatório é então subtraído das posses do jogador, e essa dedução aparece no saldo total.

A seguir é oferecida a opção de iniciar a escavação — a linha 2110 pergunta: "DESEJA COMEÇAR AS ESCAVAÇÕES (S/N)?" Se a resposta for sim, o programa prosseguirá na subrotina de escavação da linha 3000.

A ESCAVAÇÃO

```
3000 BORDER 6: PAPER 6: INK 1:
CLS
3010 IF r(m)=0 THEN  PRINT  FLA
SH 1;AT 9,6;"A mina nao foi abe
rta!": FOR z=1 TO 10: SOUND .3,
-10: NEXT z: RETURN
3020 BORDER 5: INK 0: PAPER 4:
CLS
3022 PRINT  PAPER 5;TAB 14;CHR$
 147;CHR$ 148;CHR$ 149;TAB 14;C
HR$ 150;CHR$ 151;CHR$ 152;CHR$
153;TAB 13;CHR$ 154;CHR$ 155;CH
R$ 156;CHR$ 157;CHR$ 158;TAB 31
;CHR$ 32
3025 FOR z=1 TO 32: PRINT CHR$
144;: NEXT z
3060 PRINT AT 4,0;: FOR z=100 T
O 1400 STEP 100: PRINT TAB 4-LE
N STR$ z;z: NEXT z
3090 LET a(m,2)=a(m,2)-a(m,4):
LET a(m,1)=a(m,1)-a(m,4): LET a
(m,6)=a(m,6)+200: PAUSE 30
3100 PRINT AT 3,15;CHR$ 146: FO
R f=4 TO (a(m,6)/100)+3: PRINT
AT f,15;CHR$ 145: FOR w=1 TO 10
: SOUND .01,-20: NEXT w: NEXT f
3120 IF a(m,6)=c(m,4) AND c(m,5
)=1 THEN  GOTO 3500
3130 PRINT  FLASH 1; PAPER 5;AT
 6,6;" Nada de ouro ainda!": IF
 a(m,6)=c(m,2)+200 THEN  PRINT
 FLASH 1; PAPER 1; INK 6;AT 18,
0;" Nao ha ouro nesta mina.
     Va cavar em outro lugar.":
 FOR z=1 TO 10: SOUND .5,-20: N
EXT z: LET a(m,6)=0: LET r(m)=0
3140 PAUSE 150
3300 RETURN
3500 PRINT  PAPER 6; INK 2; FLA
SH 1;AT f,12;"O U R O": FOR z=-
20 TO 50: SOUND .017,z: NEXT z:
 PAUSE 75
3550 LET a(m,5)=a(m,5)+1: LET a
(m,3)=a(m,3)+c(m,3): LET a(m,1)
=a(m,1)+(a(m,3)*er): LET a(m,6)
=0: LET r(m)=0: GOTO 3300
```

```
3000 CLS:GOSUB 1510
3010 IF R(M)=0 THEN PRINT:PRINT
:PRINT "A mina não foi aberta !
":PLAY "T25506V15AF":FOR Z=1 TO
 3000:RETURN
3020 LOCATE 14,3:PRINT CHR$(195
);CHR$(196);CHR$(197):LOCATE 14
,4:PRINT CHR$(198);CHR$(199);CH
R$(200);CHR$(201)
3022 LOCATE 13,5:PRINT CHR$(202
);CHR$(203);CHR$(204);CHR$(205)
;CHR$(206):LOCATE 31,5:PRINT CH
R$(32)
3025 LOCATE 0,6:FOR Z=0 TO 29:P
RINT CHR$(192);:NEXT
3060 LOCATE 0,7:FOR Z=100 TO 14
00 STEP 100:PRINT TAB(5-LEN(STR
$(Z)));Z:NEXT Z
3090 A(M,1)=A(M,1)-A(M,3):A(M,0
)=A(M,0)-A(M,4):A(M,5)=A(M,5)+2
00
3100 LOCATE 15,3:PRINT CHR$(194
):FOR F=4 TO (A(M,5)/100)+5:LOC
ATE 15,F:PRINT CHR$(193):PLAY"T
25501S12M1V15A64":NEXT F
3120 IF A(M,5)=C(M,3) AND C(M,4
)=1 THEN 3500
3130 LOCATE 4,0:PRINT " NADA DE
 OURO AINDA!":IF A(M,5)=C(M,1)+
200 THEN LOCATE 0,21:PRINT "Não
 há ouro nesta mina":PRINT "Vá
cavar em outro lugar":PLAY"T255
03CDEFG":A(M,5)=0:R(M)=0
3140 FOR Z=1 TO 1500:NEXT
3300 RETURN
3500 LOCATE 10,0:PRINT " O U R
O"
3510 FOR Z=1 TO 10:PLAY"T25503C
A":NEXT
3550 A(M,4)=A(M,4)+1:A(M,2)=A(M
,2)+C(M,2):A(M,0)=A(M,0)+(A(M,2
)*ER):A(M,5)=0:R(M)=0:GOTO 3140
```

```
3000  HOME
3010  IF R(M) = 0 THEN  VTAB 1:
 PRINT "A MINA NAO FOI ABERTA !
": CALL  - 198: FOR Z = 1 TO 10
00: NEXT : RETURN
3020  HGR : HCOLOR= 3: HPLOT 0,
50 TO 279,50
3030  HPLOT 125,45 TO 130,45 TO
 130,40 TO 125,40 TO 125,45: HP
LOT 145,45 TO 150,45 TO 150,40
TO 145,40 TO 145,45: HPLOT 135,
50 TO 135,40 TO 140,40 TO 140,5
0
3040  HPLOT 120,50 TO 120,40 TO
 130,30 TO 135,30 TO 135,25 TO
140,25 TO 140,30 TO 150,30 TO 1
50,5 TO 175,5 TO 175,35 TO 185,
45 TO 185,50 TO 210,50 TO 175,5
3050  HCOLOR= 6: FOR I = 58 TO
151 STEP 8: HPLOT 0,I TO 100,I:
 NEXT
3090 A(M,1) = A(M,1) - A(M,3):A
(M,0) = A(M,0) - A(M,4):A(M,5)
= A(M,5) + 200
3100  HCOLOR= 5: FOR I = 5 TO (
A(M,5) * 8 / 100) + 50: HPLOT 1
55 + 4 *  RND (1),I TO 170 - 4
*  RND (1),I:X =  PEEK ( - 1633
6): NEXT
3120  IF A(M,5) = C(M,3) AND C(
M,4) = 1 THEN 3500
3130  VTAB 22: HTAB 1: PRINT "N
ADA DE OURO AINDA !": IF A(M,5)
 = C(M,1) + 200 THEN  PRINT "NA
O HA OURO NESTA MINA": PRINT "V
A CAVAR EM OUTRO LUGAR":A(M,5)
= 0:R(M) = 0
3140  FOR Z = 1 TO 1500: NEXT
3300  HGR : HOME : TEXT : RETUR
N
3500  HOME : VTAB 22: HTAB 15:
PRINT "O U R O !!!!": CALL  - 1
98
```

```
3550 A(M,4) = A(M,4) + 1:A(M,2)
 = A(M,2) + C(M,2):A(M,0) = A(M
,0) + A(M,2) * ER:A(M,5) = A(M,
5) - 0:R(M) = 0: GOTO 3140
```

```
3000 CLS
3010 IF R(M)=0 THEN PRINT @66,"
A MINA NAO FOI ABERTA!";:FOR Z=
1 TO 10:SOUND 120,1:NEXT:RETURN
3015 IF A(M,5)>0 THEN LINE(140,
40)-(157,191),PRESET,BF:GOTO 30
90
3020 PCLS:SCREEN 1,0:COLOR 3:LI
NE(0,0)-(255,31),PSET,BF
3022 PUT(131,8)-(168,31),H,PSET
3025 FOR Z=0 TO 31:PUT(Z*8,32)-
(Z*8+7,34),T,PSET:NEXT
3060 FOR Z=100 TO 1400 STEP 100
:Z$=STR$(Z)+"-":DRAW"C4S8BM"+ST
R$(49-8*LEN(Z$))+","+STR$(32+10
*Z/100):GOSUB 9000: NEXT
3090 SCREEN 1,0:A(M,1)=A(M,1)-A
(M,3):A(M,0)=A(M,0)-A(M,4):A(M,
5)=A(M,5)+200
3100 PUT (145,32)-(152,39),D,PS
ET:FOR F=4 TO (A(M,5)/100)+3:PU
T(145,F*10)-(152,F*10+9),B,PSET
:PLAY"T5001BDBDEBDBDE":NEXT
3120 IF A(M,5)=C(M,3) AND C(M,4
)=1 THEN 3500
3125 FOR Z=1 TO 1000:NEXT
3130 PRINT @40,"NADA DE OURO AI
NDA!  ";:IF A(M,5)=C(M,1)+200 T
HEN PRINT @128,"NAO HA OURO NES
TA MINA. VA CAVAR EM OUTRO LUGA
R ";:PLAY"T5003CDEFG":A(M,5)=0:
R(M)=0
3140 FOR Z=1 TO 2500:NEXT
3300 RETURN
3500 F=40+A(M,5)/10:COLOR 2:LIN
E(140,F)-(157,F+5),PSET,BF
3510 FOR Z=1 TO 10:PLAY"T502CA"
:PUT(140,F)-(157,F+5),H,NOT:NEX
T
3520 FOR Z=1 TO 2000:NEXT
3550 A(M,4)=A(M,4)+1:A(M,2)=A(M
,2)+C(M,2):A(M,0)=A(M,0)+(A(M,2
)*ER):A(M,5)=0:R(M)=0:GOTO 3300
```

Essa rotina é chamada de dois pontos diferentes do programa. A opção de cavar é oferecida após o levantamento geológico; mas ela também pode ser feita a partir do menu, aprofundando a mina em 200 m — opção 3.

Como de costume, a primeira linha limpa a tela (no Spectrum e no MSX, as cores mudam). A linha 3010 verifica se já foi feito um levantamento geológico. As linhas 3020 a 3090 cuidam dos gráficos que representam a mina. A linha 3100 ilustra o processo de escavação, ao mesmo tempo que produz alguns efeitos sonoros.

A linha 3120 verifica se as escavações atingiram o nível provável do filão e se existe ouro na mina (é possível que se tenha cavado mais de 1000 metros apenas para descobrir que não há nenhum metal). Se for achado ouro, o programa vai para a linha 3500, que dá as boas novas ao dono da mina, junto com alguns efeitos sonoros (menos no Apple e TK-2000). A linha 3550 ajusta as posses do jogador conforme a quantidade de ouro descoberta.

Se não houver ouro na mina, o programa continuará na linha 3130. Se a escavação ultrapassar o nível em que se esperava achar o metal (200 m), o jogador será informado de sua inexistência. Se o buraco ainda não chegou a essa profundidade, o jogador será informado: "NADA DE OURO AINDA!"

VENDA O OURO

```
4000 PAPER 6: INK 1: BORDER 6:
CLS
4020 PRINT  INVERSE 1;AT 2,4;"
CAIXA ECONOMICA FEDERAL ";AT 4,
5;"Agencia de Serra Pelada": PR
INT AT 8,1;" Cotacao do ouro: "
;er;" por kg";AT 14,3;"Quantos
quilos vai vender?": INPUT nte
4070 IF nte>a(m,3) THEN  PRINT
 FLASH 1;AT 18,4;"Voce nao tem
tanto ouro!"
4080 LET nte=INT nte
4090 IF nte>a(m,3) OR nte<0 THE
N  GOTO 4020
4095 PRINT AT 16,0;CHR$ 32;TAB
31;CHR$ 32
4100 LET a(m,3)=a(m,3)-nte: LET
 a(m,2)=a(m,2)+(nte*er): LET a(
m,1)=a(m,1)+(nte*er)
4130 PRINT  PAPER 5;AT 16,1;nte
;"kg vendidos por $";nte*er: PA
USE 170: RETURN
5000 RETURN
```

```
4000 GOSUB 1500
4010 LOCATE 2,0:PRINT " CAIXA E
CONOMICA FEDERAL":LOCATE 2,1:PR
INT" Agência de Serra Pelada"
4020 PRINT:PRINT "Cotação do ou
ro:":PRINT ER;"por kg de ouro":
PRINT:INPUT "Quantos kg vai ven
der";NT
4080 NT=INT(NT)
4090 IF NT>A(M,2) OR NT<0 THEN
4000
4100 A(M,2)=A(M,2)-NT:A(M,1)=A(
M,1)+NT*ER:A(M,0)=A(M,0)-NT*ER
4130 PRINT:PRINT NT;"kg vendido
s por";NT*ER:FOR Z=1 TO 2000:NE
XT
5000 RETURN
```

```
4000  HOME
4010  VTAB 1: HTAB 6: INVERSE :
 PRINT " CAIXA ECONOMICA FEDERA
L ": VTAB 2: HTAB 6: PRINT " AG
ENCIA DE SERRA PELADA ": NORMAL

4020  PRINT : PRINT "COTACAO DO
 OURO:": PRINT ER;" POR KG DE O
URO": PRINT : INPUT "QUANTOS KG
 VAI VENDER ? ";NT
4080 NT =  INT (NT)
4090  IF NT > A(M,2) OR NT < 0
THEN 4000
4100 A(M,2) = A(M,2) - NT:A(M,1
) = A(M,1) + NT * ER:A(M,0) = A
(M,0) + NT * ER
4130  PRINT : PRINT NT;" KG VEN
DIDOS POR ";NT * ER: FOR Z = 1
TO 1000: NEXT : RETURN
5000  RETURN
```

```
4000 CLS
4020 PRINT @4,"CAIXA ECONOMICA
FEDERAL":PRINT @136,"COTACAO DO
 OURO ":PRINT @197,"1 KG DE OUR
O=";ER:PRINT @228,"QUANTOS KG V
AI VENDER";:INPUT NT
4080 NT=INT(NT)
4090 IF NT>A(M,2) OR NT<0 THEN
4020
4100 A(M,2)=A(M,2)-NT:A(M,1)=A(
M,1)+(NT*ER):A(M,0)=A(M,0)+(NT*
ER)
4130 PRINT @448,NT;"KG VENDIDOS
 POR ";NT*ER:FOR Z=1 TO 1000:NE
XT:RETURN
5000 RETURN
```

A linha 4000 limpa a tela, como de costume. A linha 4020 coloca um rótulo no vídeo, imprime a cotação do ouro no mercado e pergunta quantos quilos serão vendidos. A linha 4070 verifica se o jogador tem realmente a quantidade de metal que pretende vender. A linha 4080 certifica-se de que essa quantidade é um número inteiro.

Se o jogador tentar vender mais do que tem, ou quiser comercializar uma quantidade negativa de ouro, a linha 4090 enviará o programa de volta ao início da sub-rotina. A linha 4100 ajusta as posses do jogador de acordo com o ouro vendido.

Finalmente, a sub-rotina informa ao jogador, por intermédio da linha 4130, a quantia que ele recebeu pela venda do ouro. A linha 5000 apresenta a opção de passar a vez.

RETOQUES FINAIS

```
   1 FOR n=USR "a" TO USR "o"+7
   : READ a. POKE n,a: NEXT n
7000 PAPER 5: INK 0: BORDER 5:
CLS
7010 PRINT AT 2,4;"JORNAL DE SE
RRA PELADA";AT 4,3;"Falencias
e Concordatas"
```

```
7020 PRINT AT 10,0;"   Faliu ho
je a empresa de mine-racao ";a$
(m);" Ltda,": PRINT "gracas a m
a administracao de seuproprieta
rio."
7025 PRINT  FLASH 1;AT 20,1;" Q
ualquer tecla para recomecar "
7030 PAUSE 0: RUN 5
8000 DATA 255,85,170,0,0,0,0,0,
62,28,56,126,28,62,120,28
8010 DATA 255,255,62,126,127,60
,124,126,0,0,0,0,1,1,1,1
8020 DATA 7,29,49,45,255,255,91
,126,128,96,48,80,152,140,252,1
38
8030 DATA 1,1,1,49,49,49,49,255
,122,187,62,95,153,255,153,126
8040 DATA 209,177,224,128,128,1
28,128,128,0,0,128,128,64,32,32
,16
8050 DATA 1,3,7,7,4,4,7,7,255,2
55,255,255,149,149,159,159
8060 DATA 24,126,153,255,126,15
3,126,219,128,192,224,240,248,1
68,248,255
8070 DATA 16,8,8,4,14,31,31,255
```

```
10 R=RND(-TIME)
70 SCREEN 1:COLOR 1,10,4:KEY OF
F
80 FOR I=0 TO 119
90 READ A:VPOKE BASE(7)+192*8+I
,A:NEXT
7000 GOSUB 1500
7010 LOCATE 2,0:PRINT" JORNAL D
E SERRA PELADA ":LOCATE 3,4:PRI
NT" FALENCIAS E CONCORDATAS"
7020 PRINT:PRINT "Faliu hoje a
emprêsa de":PRINT "mineração ";
A$(M);" Ltda.":PRINT "graças à
má administração":PRINT "de seu
 proprietário"
7030 LOCATE 0,19:PRINT "QUALQUE
R TECLA PARA RECOMEÇAR"
7040 IF INKEY$="" THEN 7040 ELS
E RUN
8000 DATA 255,85,170,0,0,0,0,0,
62,28,56,126,28,62,120,28
8010 DATA 255,255,62,126,127,60
,124,126,0,0,0,0,1,1,1,1
8020 DATA 7,29,49,45,255,255,91
,126,128,96,48,80,152,140,252,1
38
8030 DATA 1,1,1,49,49,49,49,255
,122,187,62,95,153,255,153,126
8040 DATA 209,177,224,128,128,1
28,128,128,0,0,128,128,64,32,32
,16
8050 DATA 1,3,7,7,4,4,7,7,255,2
55,255,255,149,149,159,159
8060 DATA 24,126,153,255,126,15
3,126,219,128,192,224,240,248,1
68,248,255
8070 DATA 16,8,8,4,14,31,31,255
```

```
7000  HOME
7010  VTAB 1: HTAB 7: INVERSE :
 PRINT " JORNAL DE SERRA PELADA
 ": NORMAL : VTAB 3: HTAB 7: PR
INT "FALENCIAS E CONCORDATAS"
7020  PRINT : PRINT "FALIU HOJE
 A EMPRESA DE MINERACAO": PRINT
 A$(M);" LTDA. GRACAS A MA ADMI
NISTRACAO": PRINT "DE SEU PROPR
IETARIO"
7030  PRINT : PRINT "QUALQUER T
ECLA PARA JOGAR NOVAMENTE"
7040  GET R$: RUN
```

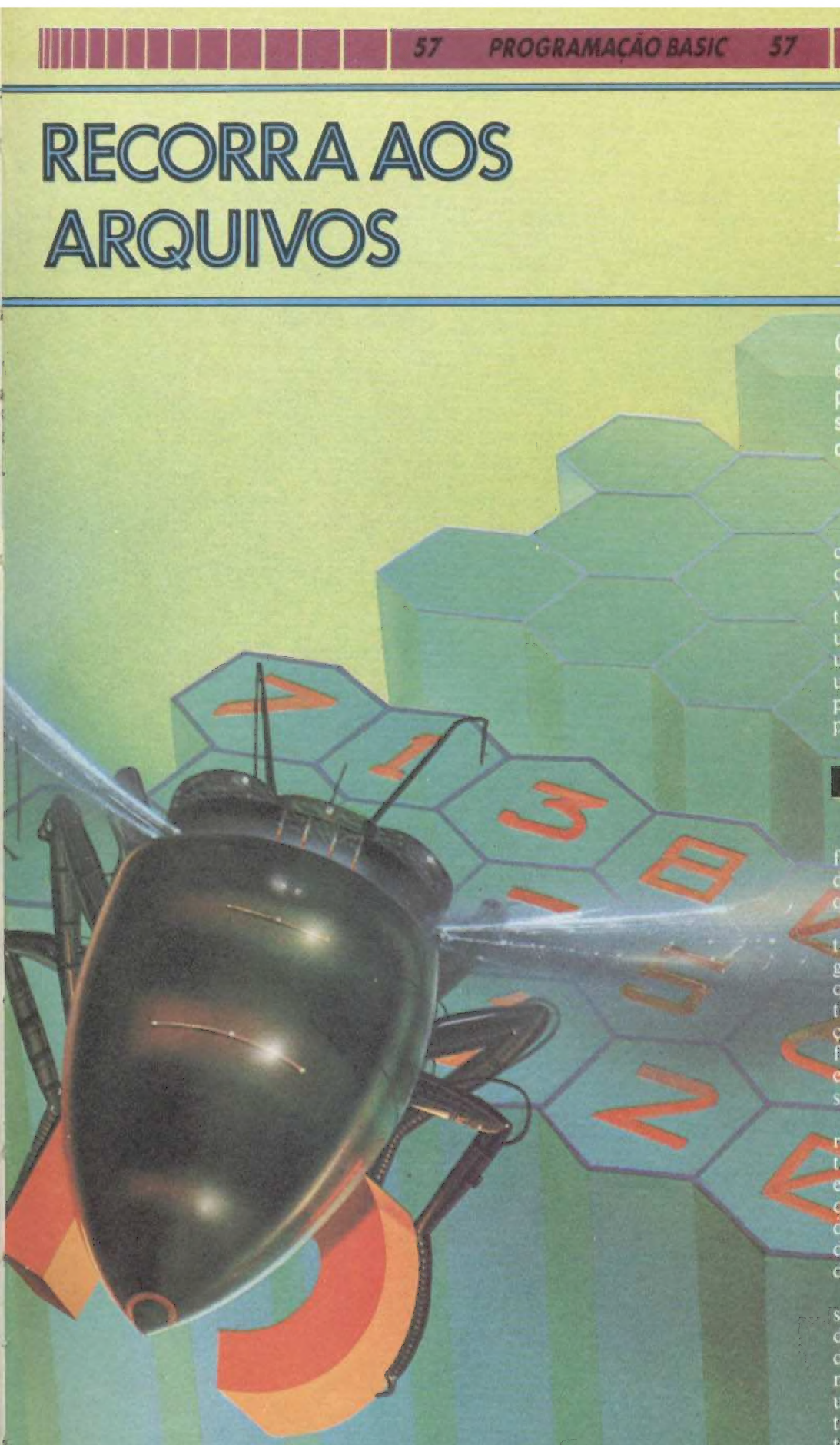
```
7000 CLS
7010 PRINT @76,A$(M):PRINT @168
,"ACABA DE FALIR! ":PRINT @449,
"QUALQUER TECLA PARA RECOMECAR"
7020 IF INKEY$="" THEN 7020 ELS
E RUN
9000 FOR K=1 TO LEN(Z$)
9010 B$=MID$(Z$,K,1)
9020 IF B$>="0" AND B$<="9" THE
N DRAW NU$(VAL(B$)):GOTO 9050
9030 IF B$="-"THEN DRAW "BF2R4"
9050 NEXT
9060 RETURN
```

As linhas 7000 a 7030 abrigam uma rotina para iniciar uma nova partida.

No Spectrum e no MSX, as linhas 8000 a 8070 contêm os dados para os blocos gráficos que desenham a mina e o processo de perfuração. No MSX, é usada a tela de textos de 32 colunas. Os padrões são guardados na tabela de padrões — **BASE(7)**. No TRS-Color, as linhas 9000 a 9060 desenham o buraco, no lado esquerdo da mina.

RECORRA AOS ARQUIVOS

- PROGRAMAS E ARQUIVOS DE DADOS
- LIMITAÇÕES DA FITA CASSETE
- TRANSFERÊNCIA DE DADOS ENTRE PROGRAMAS



Coleção de registros, dados e informações inter-relacionados do ponto de vista lógico, os arquivos são tratados em computação como se fossem uma só unidade.

Arquivo é a palavra usada para descrever qualquer tipo de conjunto de dados armazenados de uma forma acessível ao computador. A rigor, entretanto, até mesmo um programa BASIC, uma rotina em linguagem de máquina, um texto produzido por um editor ou uma informação armazenada por um programa que gerencia bancos de dados podem ser considerados arquivos.

ARQUIVOS SERIAIS

Os arquivos são normalmente classificados segundo suas características ou de acordo com os objetivos para os quais estão voltados.

O mais comum e mais simples dentre eles é o *serial*, composto de dados gravados e lidos sempre na mesma seqüência. Esse tipo de arquivo é constituído de três partes básicas: um "cabeçalho" que, entre outras coisas, identifica o arquivo; uma "fileira" de dados e, finalmente, um marcador que define seus limites.

Uma das desvantagens do arquivo serial é que as informações que ele contém só podem ser processadas na ordem em que foram gravadas. Isso significa que, se o seu programa procurar por um dado que está no meio do arquivo, todo o conjunto deverá ser pesquisado desde o começo.

Tal situação, contudo, pode ser sensivelmente melhorada se os dados forem distribuídos com certa ordem. Quase todo arquivo usado em aplicações é ordenado alfanumericamente. Desse modo, um arquivo serial pode ser corretamente chamado de *seqüencial*, embora tal termo seja mais usado para definir qualquer espécie de arquivo gravado em série. Um verdadeiro arquivo seqüencial tem sua estrutura definida pelo usuário ou pelo programador.

ARQUIVOS EM DISQUETE

A maior restrição para arquivos em fita cassete é que eles só podem ser lidos e gravados de forma serial; assim, eles funcionam na prática como um arquivo seqüencial. Felizmente, para as aplicações mais simples, isso não chega a ser um grande problema.

Os arquivos seriais são usados com muita freqüência, não somente em sistemas que funcionam apenas com fitas cassete, como também naqueles que empregam métodos versáteis de armazenamento em disquete.

Por outro lado, a maneira com que se lêem dados em disco permite que se utilize um outro tipo de arquivo: o *arquivo de acesso randômico* (também chamado de arquivo de acesso direto ou arquivo relativo). Os sistemas que trabalham com disco empregam ainda outros tipos de arquivo. Mas estes ou não são usados por micros domésticos ou são específicos de um determinado sistema operacional de disco.

Todo disquete conta com um certo número de blocos, onde são armazenadas as informações. Esses blocos se parecem com células de um favo de mel: ligadas entre si, mas claramente separadas. Os blocos são identificados por coordenadas: trilhas e setores. Isto permite que eles possam ser acessados diretamente e em qualquer ordem por um determinado programa.

Cada bloco pode ser lido, corrigido, regravado, sem que os outros sofram interferências. Desse modo, o processamento da informação se dá muito rapidamente e o tamanho do arquivo é limitado apenas pela capacidade de armazenamento do disco em uso.

ARQUIVOS NO FORMATO ASCII

Quando a informação arquivada é independente de programas e encontra-se armazenada separadamente, a quantidade de arquivos pode ser quase infinita. Com um programa editor de textos, por exemplo, tem-se arquivos separados para cartas padrão, artigos, avisos etc.

Uma das maiores vantagens de se trabalhar com um sistema de arquivos separados é que vários programas podem usar a mesma fonte de dados e não necessariamente do mesmo computador. Uma planilha de cálculo pode acessar arquivos de outros programas e um banco de dados padrão pode ser usado para armazenar informações destinadas a qualquer tipo de aplicação.

Isso pode ser feito por meio de arquivos seqüenciais escritos na liguagem padrão do código ASCII (para maiores esclarecimentos, veja o artigo *Trabalhe com o Código ASCII*, à página 361). A maioria dos computadores (mesmo aqueles que usam um código um pouco diferente do ASCII) permite que se opere com esses arquivos.

Neste caso, tudo que é gravado, incluindo comandos e variáveis, é transformado em seqüências de caracteres do código ASCII. Um programa acaba se transformando num arquivo de texto.

Uma vez gravada, a informação torna-se, pelo menos em teoria, acessível a outros computadores que possam ser ligados ao seu. Quando se usa um arquivo no formato ASCII, pode-se chamá-lo por intermédio de um editor de textos. Assim, todas as facilidades oferecidas pelo editor podem ser utilizadas para escrever o seu programa. E, quando estiver pronto, ele será gravado como um arquivo de texto.

Note que a "transportabilidade" de um arquivo no formato ASCII de uma para outras máquinas pode ser impedida se nele forem colocados caracteres de controle. Esses códigos são habitualmente usados pelo editor para controlar a impressora. E, como tais caracteres não são padronizados, provavelmente causarão problemas quando transferidos para outra máquina. Dessa forma, um programa deve ser gerado sempre com a opção "não-documentação" ou "não-formatação" (isto é, sem marcas de avanço de linha, parágrafo, negrito e outras, encontradas nos arquivos de texto a serem impressos como cartas).

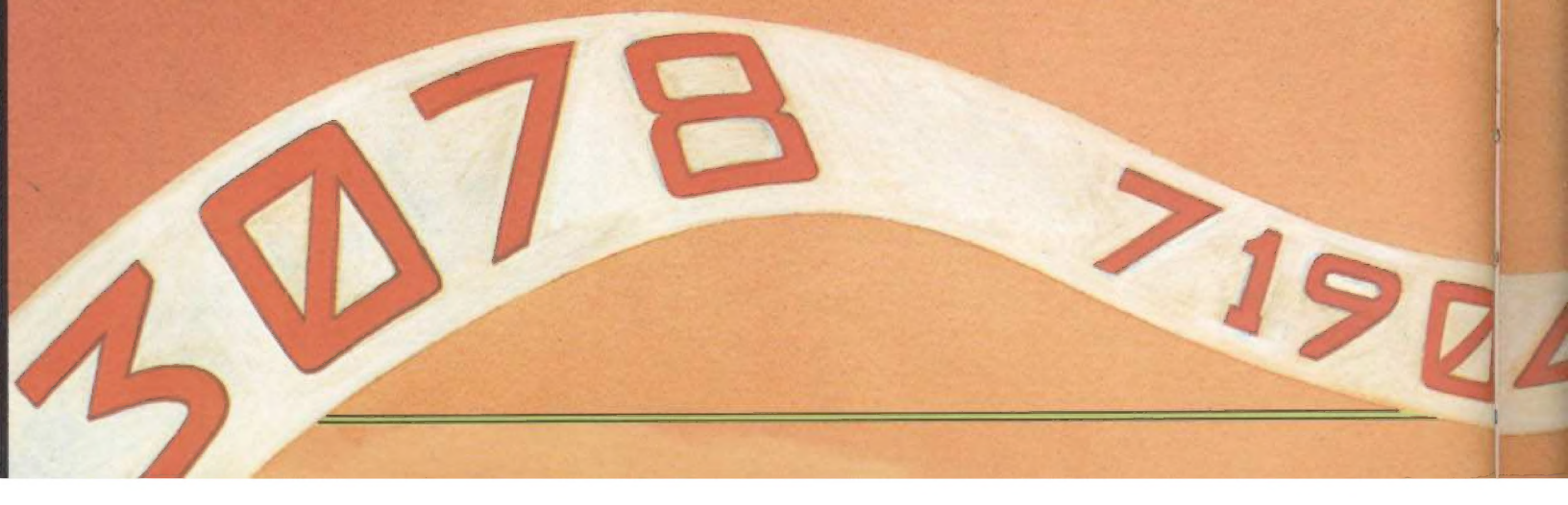
ARQUIVOS DE DADOS

Qualquer programa que utilize um banco de dados — como um gerenciador de arquivos de dados, uma planilha de cálculo, ou os pacotes de contabilidade — funciona manipulando informações já fornecidas ao sistema. Ao se trabalhar com um programa desse tipo, seria extremamente cansativo ter que digitar toda a informação sempre que um aplicativo (como um programa de mala direta) fosse usado.

A solução para esse problema consiste em armazenar a informação separadamente do programa, como um arquivo de dados. Tal procedimento aumenta muito a capacidade de programa e diminui as exigências de memória do computador, visto que um sistema de armazenamento funciona como uma memória virtual, ficando na memória do computador apenas os dados em uso no momento.

O menor componente de um sistema de arquivamento — computadorizado ou não — é, de fato, a *entrada*. Várias entradas compõem um *registro*. Cada registro é dividido em *campos*. Cada campo tem um título ou rótulo, que permanece inalterado para todos os registros daquele arquivo. Esse rótulo serve para identificar o tipo de dado que o campo contém. Pode-se ver este tipo de estrutura em operação no programa apresentado no artigo *Organize as suas Coleções* (*1*), à página 68.

Consideremos um arquivo simples de fichas. Ele é formado pelo conjunto completo das fichas, que são relaciona-



das por algum ponto comum (a lista dos sócios de um clube, por exemplo). Um registro é, neste caso, uma ficha usada para identificar apenas um membro do clube. Os campos são cada parte da informação contida na ficha — nome, endereço, telefone etc.

Em um sistema que funcione com papel, o arquivo pode ser transportado em um fichário, em pastas, caixas etc. Esses veículos, habitualmente chamados de arquivo, na realidade não passam de um modo físico de conter (e, às vezes, transportar) a informação.

Um sistema computadorizado funciona de modo bem diferente. A não ser que todos os dados estejam armazenados num só disco ou fita, não se estabelece aí a mesma relação "física" existente entre os pedaços de papel e seu invólucro. Em algumas aplicações, os dados ficam completamente separados, e podem ocupar discos ou fitas diferentes. Em outras, os dados são parte integrante do programa e não devem ser considerados um conjunto separado.

Arquivos separados podem, com freqüência, ser usados por programas que não são responsáveis por sua criação. Assim, uma informação que foi trabalhada e armazenada por uma planilha de cálculo pode ser acessada por um editor de textos. Repare que a palavra é acessada e não transferida — ou seja, o programa lê e usa a informação sem removê-la. Num sistema manual, a transferência constitui a única forma de passar informação de um arquivo para outro sem copiá-la.

NOMES DE ARQUIVOS

O que não muda, no entanto, é a necessidade de se dar um nome a um arquivo. Seria difícil, se não impossível, acessar alguma informação num sistema computadorizado sem um nome com que o programa pudesse identificar rapidamente o conjunto de dados.

Para que um arquivo computadorizado seja adequadamente acessado por um programa, é preciso que seu nome seja escolhido com cuidado. Se não houver nomes, os dados armazenados em fita podem ser lidos na forma "carregar o próximo arquivo". Mas qualquer rotina de busca necessita de um nome pa-

ra identificar os dados requeridos.

Em fita cassete pode-se ter mais de um arquivo com a mesma denominação, ou mesmo arquivos sem nenhum nome. Mas esses procedimentos geram graves problemas. Se você instruir o computador para ler a última versão de um arquivo que tem o mesmo nome da cópia anterior, ele não poderá decidir que versão deve ler e ficará com a primeira que aparecer. A solução é dar a cada cópia um nome adequado e único.

Quando se usa um sistema de disco, o procedimento para a escolha de nomes é o mesmo. Nesse sistema, porém, é perfeitamente possível gravar um arquivo no lugar de outro (apagando este último), caso o mesmo nome tenha sido dado aos dois. Alguns sistemas operacionais de disco impedem que isso aconteça, mas, geralmente, deve-se proteger o arquivo de um apagamento acidental — "travando-o", como se diz no jargão da informática.

Ao escolher o nome, certifique-se de que não está ultrapassando o tamanho máximo permitido por cada micro. Em fita, o espaço para o nome de um arquivo não vai além de dez caracteres no Spectrum, oito no TRS-Color e seis no MSX. O Apple não usa nome para arquivos em fita. Em sistemas de disco, o tamanho varia bastante.

Convém ressaltar que o nome de qualquer arquivo computadorizado deve ser de fácil memorização. Além disso, ele precisa traduzir claramente seu conteúdo. Uma lista de todos os seus arquivos também ajudará bastante.

Os processos de transferência de dados através de um mecanismo de armazenamento são descritos pelas palavras gravar e ler (*write* e *read*, em inglês). Cada computador tem um conjunto diferente de comandos para essas duas operações, mas, de um modo geral, as rotinas são muito parecidas.

Normalmente, o programa principal é o primeiro a ser carregado na memória do computador — ele pode ser, por exemplo, um editor de textos ou um gerenciador de banco de dados.

Se o programa precisar de dados que não estão nele próprio, estes devem ser carregados de um arquivo de dados ou do teclado, manualmente. A partir daí, os dados serão corrigidos, alterados e trabalhados. Eles não poderão ser modificados enquanto permanecerem no disco ou na fita.

Em alguns arquivos, toda a informação tem que ser colocada na memória, mesmo que apenas uma pequena porção dela vá ser trabalhada. Este é o caso do Spectrum com um Microdrive. Em outros casos, pode-se ler apenas os dados que serão modificados. Por fim, a informação é gravada em um mecanismo externo de armazenamento (disquete ou fita cassete), ou enviada a outro mecanismo, como uma impressora.

Analisemos agora, mais detalhadamente, os estágios de gravação e leitura. Primeiro, deve ser dada ao computador uma instrução para abrir um canal de comunicação, de modo que ele saiba que a informação transitará por aquele ponto. Outras informações podem ser fornecidas a partir do comando de abertura do canal de comunicação, para especificar, por exemplo, que tipo de periférico está sendo acessado. Geralmente o periférico será um gravador cassete ou uma unidade de disquete.

Aberto o canal, são usados comandos para enviar ou gravar a informação no periférico escolhido. Inicialmente, os comandos são empregados para receber e não para enviar dados. Em qualquer dos casos, uma área de memória será utilizada para armazenamento temporário de informações (*buffer*).

Quando o acesso à informação termina, é preciso fechar o canal de comunicação. Assim, toda informação que estiver no meio do trajeto (ou eventualmente parada no buffer) será forçada a terminar o percurso. Para encerrar, o computador marca o fim do arquivo — *end of file* (eof), em inglês.

Até ser reaberto, o arquivo não transmitirá nem receberá nenhuma outra informação. Isso não impede a manipulação de arquivos por outros canais eventualmente abertos (sempre que o sistema seja capaz de trabalhar ao mesmo tempo com vários canais).

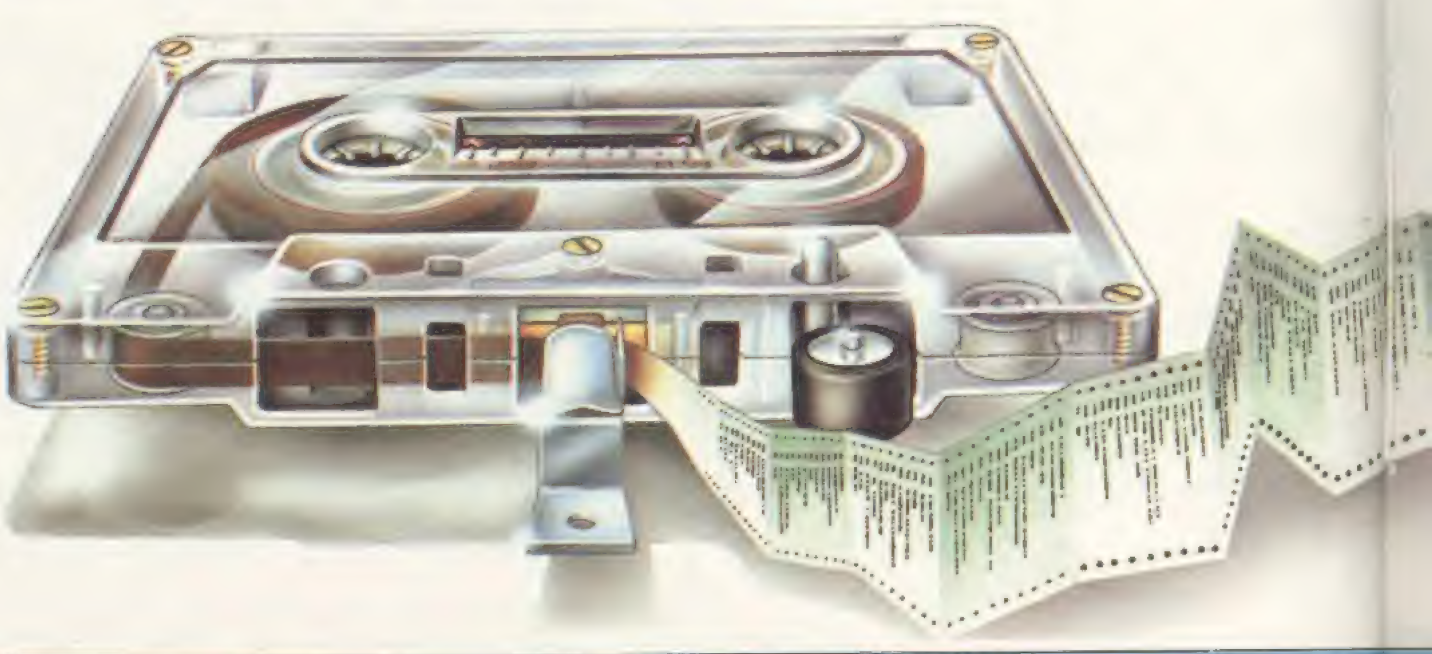
As instruções para executar essas tarefas variam conforme o micro. Vejamos cada uma delas. As instruções **SAVE** e **LOAD** (e variantes) que se aplicam a programas não serão abordadas.

Em cada um dos exemplos, a letra **N** se refere ao número do arquivo que é usado por muitos dos comandos; **X** (ou **X$**) é a informação transmitida.

S

Se você usa um gravador cassete para armazenar dados, está limitado a gravar programas, bytes e matrizes. Arquivos de dados podem ser manipulados com um Microdrive adequadamente instalado (esse dispositivo não é fabricado no Brasil). Os comandos de entrada e saída são os seguintes:

OPEN#



utilizado com a finalidade de preparar o sistema para a transferência de informação.

Em sua forma padrão, esse comando fica assim:

```
OPEN#N;"m";1;"NOME DE ARQUIVO"
```

onde a expressão **NOME DE ARQUIVO** pode ser uma variável previamente definida. Os dados, por sua vez, são transmitidos pelo canal de número **N**, sendo **m** o canal do Microdrive.

```
PRINT#
```

usado na forma **PRINT #N;X** (ou **X$**) com o objetivo de enviar a informação a ser gravada para o buffer de dados.

```
INPUT#
```

utilizado na forma **INPUT #N;X** (ou **X$**) para ler dados do buffer de dados. Estes são lidos até que surja um **CR** (*carriage return*).

```
CLOSE#
```

fecha um canal de comunicação previamente aberto. Seu formato é **CLOSE #N**.

Existem vários comandos para abrir linhas de comunicação, principalmente em um gravador cassete. Já as unidades de disco são acessadas por comandos que variam de acordo com o sistema operacional de disco em uso.

```
OPEN
```

abre um canal de comunicação para um periférico específico. Para gravar dados em fita, esse comando toma a seguinte forma:

```
OPEN "O",#-1,"NOME DE ARQUIVO"
```

onde apenas **NOME DE ARQUIVO** é uma variável definida pelo programa do usuário. O número -1 indica o gravador cassete. Se quisermos abrir um arquivo para leitura, devemos usar o comando na seguinte forma:

```
OPEN "I",#-1,"NOME DE ARQUIVO"
```

```
INPUT#
```

lê um cordão de um arquivo. A instrução **EOF(-1)** (eof significa *fim de arquivo*) pode ser usada para evitar erros (impede que se tente ler dados além da marcação de fim de arquivo).

```
PRINT#
```

envia informações para o periférico especificado pelo número **N**. O comando toma esta forma:

```
PRINT#N,X$
```

onde o caractere **N** é –1 para o gravador cassete e –2 para a impressora.
Já o comando

```
CLOSE#
```

fecha o canal de comunicação com o periférico. É usado na forma:

```
CLOSE#N
```

onde N é geralmente –1.

O MSX pode ser conectado tanto a um gravador cassete como a um controlador de discos para armazenamento de programas e dados. Abordaremos aqui os comandos referentes à manipulação de arquivos de dados em fita:

```
OPEN
```

utilizado para abrir um canal de comunicação com vários periféricos, que podem ser: gravador cassete, tela de texto, tela de gráfico, impressora, acionador de discos. Num arquivo em fita, o comando toma esta forma:

```
OPEN "CAS:NOME ARQUIVO"
FOR INPUT AS #N
```

para se ler dados de um determinado arquivo usando o canal **N** (onde **N** pode ser 1, 2 etc.). Para gravação de dados, substitui-se **INPUT** por **OUTPUT**.

```
PRINT#
```

serve para enviar dados a um periférico. Toma a forma

```
PRINT#N,X (ou X$)
```

Quando se usa mais de uma variável na mesma instrução **PRINT #**, é necessário que sejam tomados alguns cuidados. Se forem variáveis numéricas, não há problema, pois o BASIC se encarregará de separá-las. No entanto, expressões alfanuméricas devem ser separadas explicitamente por vírgulas:

```
PRINT#1,A$;",";B$
```

Esse comando pode ser combinado com a instrução **USING** para determinar o formato do dado.

```
INPUT#
```

lê dados de um determinado periférico. Toma a forma

```
INPUT#N,X (ou X$)
```

No caso de variáveis numéricas, um espaço, uma vírgula, um retorno de carro (**CR**) ou um avanço de linha (**LF**) indicam o fim do número.

Para variáveis alfanuméricas, serão reconhecidos os marcadores acima (com exceção do espaço).

```
LINE INPUT#
```

é usado para fazer a leitura de variáveis alfanuméricas de um periférico. A diferença dessa instrução em relação ao comando anterior é que apenas o retorno de carro (**CR**) marca o fim do cordão que está sendo lido.

```
EOF(N)
```

indica se o final do arquivo foi encontrado. É habitualmente usado para evitar que se tente ler dados inexistentes no arquivo. O valor –1 retorna quando o marcador de fim de arquivo é encontrado. Esse comando é mais utilizado na forma

```
IF EOF(1) THEN instrução
```

onde **instrução** provoca geralmente o fechamento do arquivo.

```
CLOSE
```

serve para fechar o canal de comunicação de número **N**. Força o envio dos dados remanescentes no buffer de dados para o periférico ligado a esse canal. Toma a forma

```
CLOSE#N
```

O Apple e o TK-2000 são extremamente limitados em sua capacidade de manipular arquivos em fita. Como no Spectrum, apenas programas e matrizes podem ser gravados em fita. A seguir, descreveremos alguns comandos e aspectos mais importantes da manipulação de arquivos pelo sistema operacional DOS #3.3 do Apple.

Inicialmente, convém chamar a atenção para um fato essencial, que já custou muitas horas de trabalho a mais de um programador: todos os comandos do sistema operacional devem ser precedidos de um caractere especial — **CTRL-D**. Esse caractere deve estar sempre na primeira posição da linha. Desse modo, se antes de um comando houver um **PRINT** seguido de um ponto e vírgula, que deixe o cursor numa posição que não seja a primeira da linha, o comando não será reconhecido como do DOS. Igualmente importantes são as formas de colocar o **CTRL-D** na linha do comando. A mais segura consiste em definir uma variável que contenha esse caractere e colocá-la na linha de comando. Outra maneira é simplesmente acionar **CCTRL-D** (ou seja, pressionar a tecla **CTRL** simultaneamente com a tecla **D**) logo após ter aberto as aspas exigidas pelos comandos do DOS.

Nos exemplos a seguir, recorreremos ao primeiro método, usando a variável **D$** (freqüente entre os programadores do Apple). Ela é definida por meio do código ASCII de **CTRL-D:**

```
D$=CHR$(4)
```

Vejamos alguns comandos:

```
OPEN
```

abre um canal de comunicação entre a CPU e uma unidade de disco, assumindo a seguinte forma:

```
PRINT D$;"OPEN NOME ARQUIVO",
Dd, Ss, Vv
```

onde **NOME ARQUIVO** pode ser representado por uma variável alfanumérica e ter até trinta caracteres. A letra **D** é seguida do número do drive acessado (1 ou 2). A letra **S** é acompanhada do número do soquete (ou *slot*) no qual a placa que controla as unidades de disco está instalada (esse número é, em geral, 6). E **V** é seguido do número do volume do disquete, que é definido no momento de sua formatação. Esses três últimos parâmetros são opcionais e independentes: a presença de um não implica o uso dos demais.

```
READ
```

é empregado para habilitar um arquivo em disco para a leitura de dados e toma a seguinte forma:

```
PRINT D$,"READ NOME ARQUIVO"
```

A seguir, digita-se o comando **INPUT** para ler os dados desejados do arquivo em uso. O comando **PRINT D$**, por sua vez, encerra a leitura de informações do arquivo de dados (**D$** deve conter o caractere **CTRL-D**).

```
WRITE
```

envia e grava os dados em um arquivo em disco. Ele é usado na forma

```
PRINT D$;"WRITE NOME ARQUIVO"
```

Ele deve ser acompanhado de vários comandos **PRINT** para o envio dos dados. No caso de variáveis alfanuméricas, elas devem ser separadas por um **CR** (retorno de carro) para serem lidas individualmente mais tarde:

```
PRINT A$;CHR$(13);B$
CLOSE
```

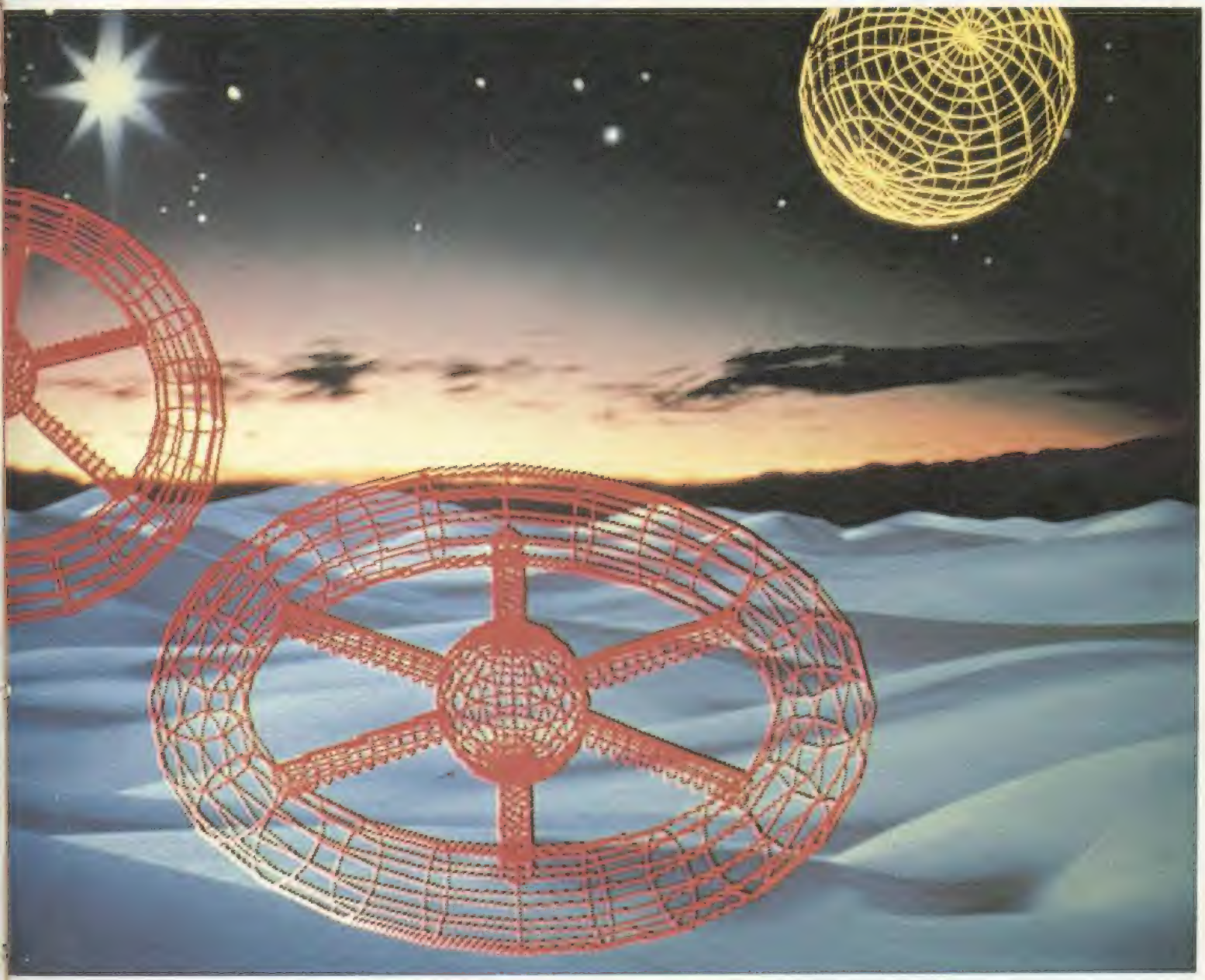
fecha o canal de comunicação estabelecido. Ele é usado na forma

```
PRINT D$,"CLOSE NOME ARQUIVO"
,Dd, Ss, Vv
```

Como no comando **OPEN**, os três últimos parâmetros são opcionais. Neste caso, **NOME ARQUIVO** pode ser omitido, provocando o fechamento de todos os arquivos abertos no momento.

PROGRAMAÇÃO DE GRÁFICOS EM 3-D (4)

- OBJETOS MÚLTIPLOS
- VARIE A PERSPECTIVA
- COMO DESENHAR UM GLOBO
- CONTORNE O OBJETO
- DESENHE UM TORO



Agora que você já conhece os segredos da perspectiva, pode reunir vários problemas e rotinas e desenhar as mais diversas figuras flutuando no espaço interestelar.

Até aqui trabalhamos apenas com formas retangulares, quadrados e círculos concêntricos. Ao avançar um pouco mais em nosso estudo, entretanto, aprendemos a combinar (por meio de várias transformações) uma série de grades numa vista em perspectiva de um cubo. Este artigo ensina como pequenas mudanças no programa desenvolvido até este ponto pode nos tornar capazes de desenhar uma grande variedade de imagens tipo *wireframe*.

Começaremos com um desenho de figuras múltiplas. Se você já testou o primeiro programa do artigo *Programação de Gráficos em 3-D* (*3*), deve ter notado as alterações da imagem em perspectiva quando a distância do observador em relação ao objeto (D) é pequena, e o pouco efeito de perspectiva quando essa distância, ao contrário, é muito grande. Esse programa será desenvolvido agora de maneira a mostrar quatro cubos e não apenas um.

MULTIPLICIDADE DE OBJETOS

Os usuários que gravaram em fita ou disco os programas dos artigos precedentes desta série estão livres de boa parte do trabalho de digitação. Os que não fizeram isso terão que inserir não só o programa do artigo anterior (que se inicia na página 641) como também a rotina que desenha círculos, apresentada no primeiro artigo desta série (página 581).

O programa é constituído de blocos e rotinas, o que facilita a sua modificação para representar ao mesmo tempo quatro imagens em vez de uma.

Desse modo, se quiséssemos desenhar um conjunto de quatro cubos, por exemplo, precisaríamos especificar suas posições na tela e chamar quatro vezes a Rotina do Cubo. A posição estaria melhor especificada na rotina que transforma as coordenadas.

Passemos da teoria à prática. Para começar, modifique essa parte do programa como mostramos a seguir (mas não rode o programa por enquanto):

```
8500 X1 = T1 * X + T4 * Y + T7
+ XO
8510 Y1 = T2 * X + T5 * Y + T8
+ YO
8520 Z1 = T3 * X + T6 * Y + T9
+ ZO
```

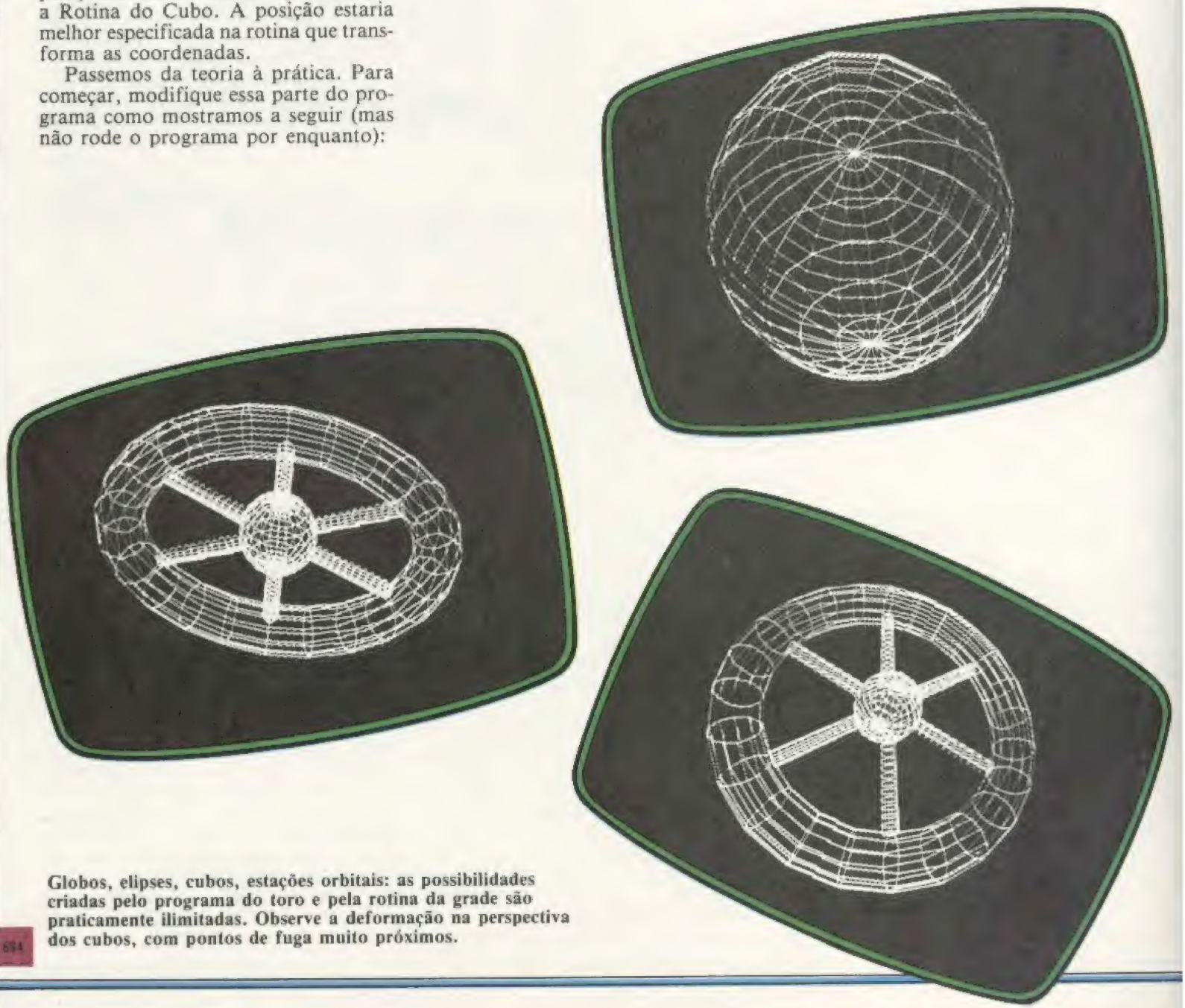
```
8500 X1=T1*X+T4*Y+T7+XO
8510 Y1=T2*X+T5*Y+T8+YO
8520 Z1=T3*X+T6*Y+T9+ZO
```

```
8500 LET X1=T1*X+T4*Y+T7+XO
8510 LET Y1=T2*X+T5*Y+T8+YO
8520 LET Z1=T3*X+T6*Y+T9+ZO
```

As variáveis **XO**, **YO** e **ZO** no final dessas três linhas especificam um equivalente, em cada uma das três coordenadas, para determinar a separação dos quatro cubos. São atribuídos valores a essas variáveis antes de os cubos serem desenhados.

Agora digite essa outra seção de programa e rode-a para ver qual é o efeito do equivalente:

```
1520  GOSUB 1000
1530 XO = PP:YO =  - PP:ZO = 0
120 L = 20:PP = 20:N = 2
150  GOSUB 1500
1500 XO =  - PP:YO =  - PP:ZO =
 0
```



Globos, elipses, cubos, estações orbitais: as possibilidades criadas pelo programa do toro e pela rotina da grade são praticamente ilimitadas. Observe a deformação na perspectiva dos cubos, com pontos de fuga muito próximos.

```
1540  GOSUB 1000
1550 XO = PP:YO = PP:ZO = 0
1560  GOSUB 1000
1570 XO =  - PP:YO = PP:ZO = 0
1580  GOSUB 1000
1590  RETURN
```

T

```
120 L=10:PP=20:N=2
150 GOSUB 1500
1500 XO=-PP:YO=-PP:ZO=0
1520 GOSUB 1000
1530 XO=PP:YO=-PP:ZO=0
1540 GOSUB 1000
1550 XO=PP:YO=PP:ZO=0
1560 GOSUB 1000
1570 XO=-PP:YO=PP:ZO=0
1580 GOSUB 1000
1590 RETURN
```

MSX

A versão para o MSX é a mesma do TRS-Color, exceto pela linha:

```
120 L=20:PP=20:N=2
```

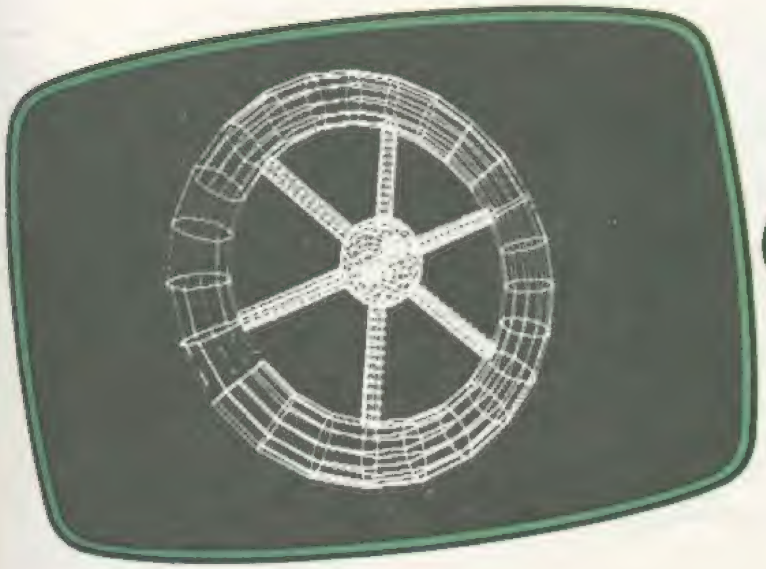
S

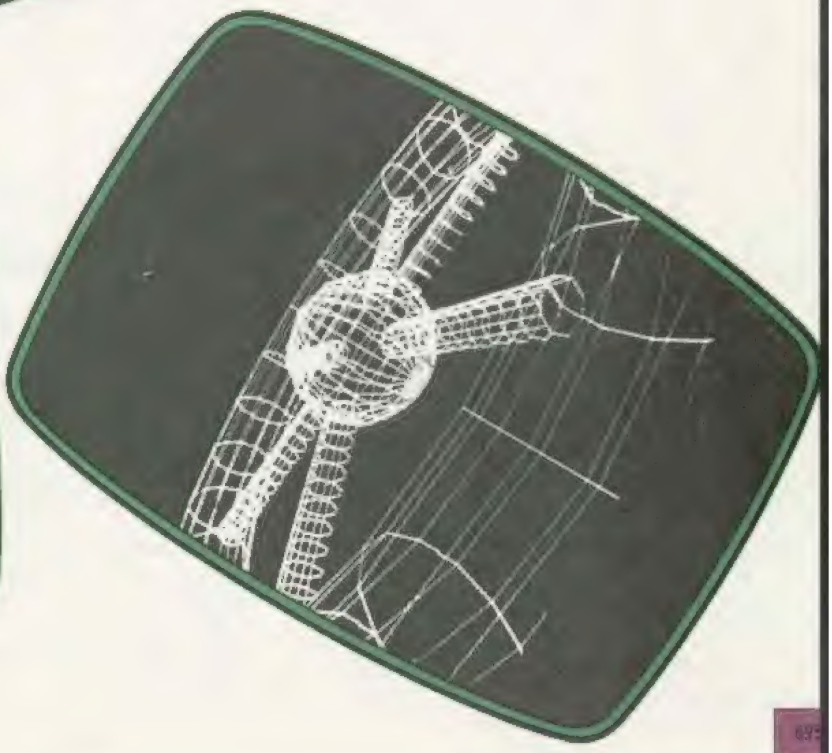
```
 120 LET L=20: LET PP=20: LET N
 =1
 150 GOSUB 1500
1500 LET XO=-PP: LET YO=-PP: LE
T ZO=0
1520 GOSUB 1000
1530 LET XO=PP: LET YO=-PP: LET
 ZO=0
1540 GOSUB 1000
1550 LET XO=PP: LET YO=PP: LET
ZO=0
1560 GOSUB 1000
1570 LET XO=-PP: LET YO=PP: LET
 ZO=0
1580 GOSUB 1000
1590 RETURN
```

A linha 120 contém a variável **PP**, que transfere o valor para o equivalente na linha 1500.

A linha 150 chama uma rotina para desenhar quatro cubos. A sub-rotina especifica uma posição, como na linha 1530, por exemplo, e chama a rotina do cubo na linha seguinte, desenhando assim os quatro cubos.

Ao rodarmos o programa, teremos que fornecer alguns dados. O programa pedirá primeiro pelo valor de **D** (distância do plano de projeção). O valor 1000 será suficiente para começarmos. O programa então pedirá para que forneçamos a posição do observador (X, Y e Z). Experimente os valores que sugerimos a seguir e note como as imagens se distorcem quando **D** é pequeno (mas não quando ele é grande). Tente usar seus próprios valores também, incluindo milhares para **D** e números negativos ou zero para X, Y, e Z:





D	X	Y	Z
100	55	0	0
900	200	-250	200
20000	1000	3000	10000

A grande variedade de vistas possíveis com este programa nos permite fazer vários experimentos, mas não devemos nos limitar a modificar apenas os valores da posição do observador. A linha 120 contém variáveis responsáveis pelo comprimento dos lados do cubo **L**, pela separação destes (**PP**) e pelo número de quadrados na grade (**N**): três variáveis que podem ser alteradas antes de se rodar o programa.

DESENHE UM GLOBO

Embora curto, o programa que acabamos de inserir é capaz de desenhar imagens complexas, pois pode chamar qualquer uma das muitas rotinas contidas na listagem. Uma dessas rotinas poderia ser aquela que desenha círculos concêntricos (linhas 6000 a 6160), apresentada no artigo *Programação de Gráficos em 3-D (I)*, à página 581. Caso ela não esteja em seu programa, digite-a novamente e grave-a em disco ou fita para eventual uso futuro.

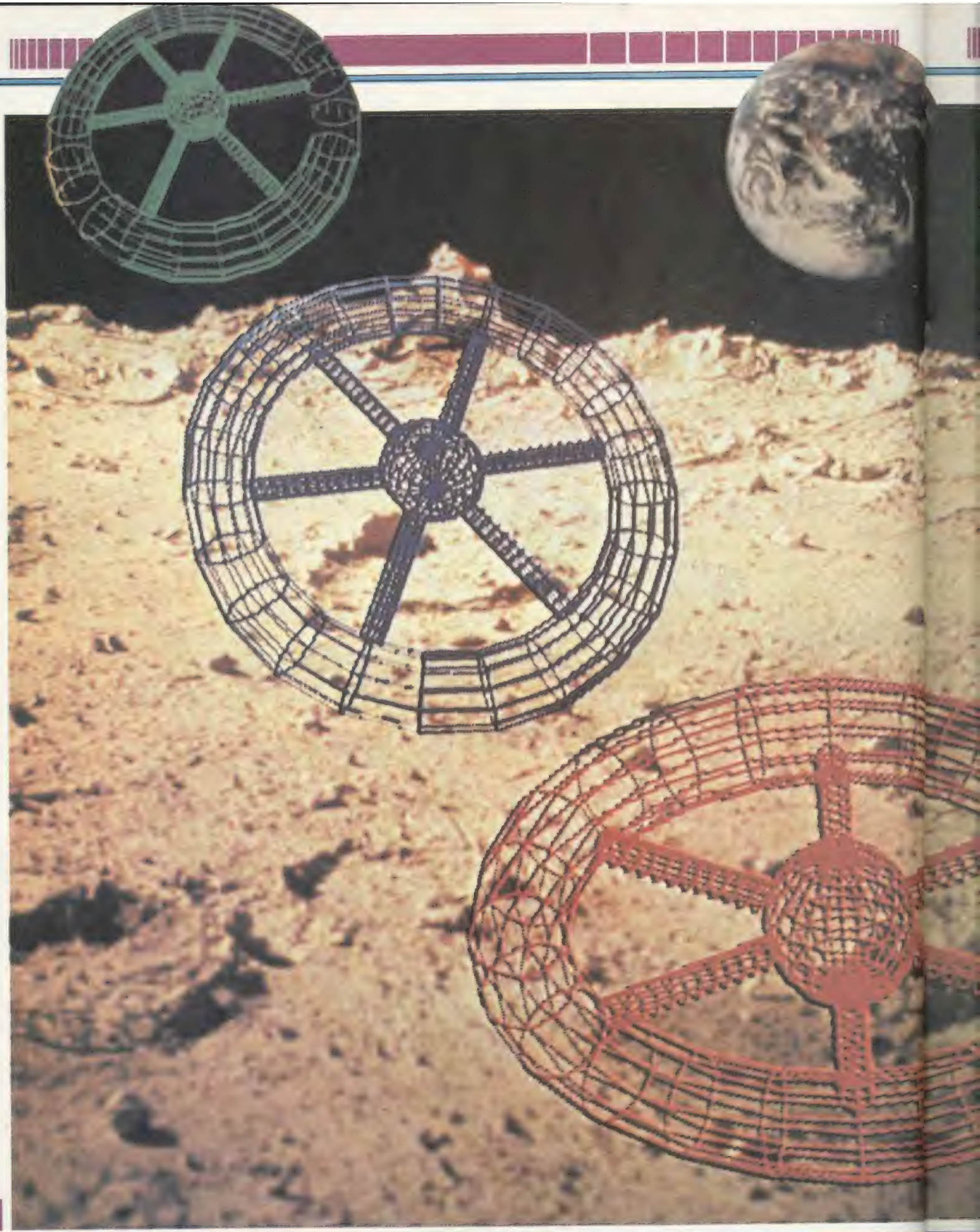
Em seguida, modificaremos o programa para que ele chame a rotina do círculo, em vez de chamar as rotinas da grade, do lado ou do cubo. Apague a linha 120 e acrescente estas linhas:

```
150 R = 20:N = 18: GOSUB 2000
2000 T7 = 0:T8 = 0:T9 = 0
2010 T4 = 0:T5 = 0:T6 = 1
2040 KA = 2 * PI / N
2050 KB = 0
2060  FOR KC = 1 TO  INT (N / 2
)
2070 T1 =  COS (KB):T2 =  SIN (
KB):T3 = 0
2080 XS = 0:YS = 0: GOSUB 6000
2090 KB = KB + KA
2100  NEXT KC
2110 T1 = 1:T2 = 0:T3 = 0
2120 T4 = 0:T5 = 1:T6 = 0
2130 KA = KA / 2
2140 KB = 0:RR = R
2150  FOR KC = 1 TO N
2160 T7 = 0:T8 = 0:T9 = RR *  C
OS (KB)
2170 XS = 0:YS = 0:R = RR *  SI
N (KB): GOSUB 6000
2180 KB = KB + KA
2190  NEXT KC
2200  RETURN
```

```
150 R=20:N=18:GOSUB 2000
2000 T7=0:T8=0:T9=0
2010 T4=0:T5=0:T6=1
2040 KA=2*PI/N
2050 KB=0
2060 FOR KC=1 TO INT(N/2)
2070 T1=COS(KB):T2=SIN(KB):T3=0
2080 XS=0:YS=0:GOSUB 6000
2090 KB=KB+KA
2100 NEXT KC
2110 T1=1:T2=0:T3=0
2120 T4=0:T5=1:T6=0
2130 KA=KA/2
2140 KB=0:RR=R
2150 FOR KC=1 TO N
2160 T7=0:T8=0:T9=RR*COS(KB)
2170 XS=0:YS=0:R=RR*SIN(KB):GOS
UB 6000
2180 KB=KB+KA
2190 NEXT KC
2200 RETURN
```

```
100 LET XO=0: LET YO=0: LET ZO
=0
150 LET R=20: LET N=18: GOSUB
2000
2000 LET T7=0: LET T8=0: LET T9
=0
2010 LET T4=0: LET T5=0: LET T6
=1
2040 LET KA=2*PI/N
2050 LET KB=0
2060 FOR K=1 TO INT (N/2)
2070 LET T1=COS KB: LET T2=SIN
KB: LET T3=0
2080 LET XS=0: LET YS=0: GOSUB
6000
2090 LET KB=KB+KA
2100 NEXT K
2110 LET T1=1: LET T2=0: LET T3
=0
2120 LET T4=0: LET T5=1: LET T6
=0
2130 LET KA=KA/2
2140 LET KB=0: LET RR=R
2150 FOR K=1 TO N
2160 LET T7=0: LET T8=0: LET T9
=RR*COS KB
2170 LET XS=0: LET YS=0: LET R=
RR*SIN KB: GOSUB 6000
2180 LET KB=KB+KA
2190 NEXT K
2200 RETURN
```

Ao rodar o programa, devemos fornecer valores para **D**, bem como para a posição do observador. Com esses valores, a linha 150 chama a rotina que acabamos de inserir e desenha um globo. Este é obtido por intermédio do de-



senho de uma série de elipses (ou círculos, dependendo da posição do observador). Tais elipses formam as linhas de longitude (que ligam os pólos do globo) e as de latitude, que correm paralelas à linha do equador.

O raio do globo é especificado em **R**, na linha 150, e o número de linhas de latitude e longitude, em **N**; este deve ser um valor par e maior do que 2. As linhas 2000 a 2050 ajustam as variáveis para posicionar o globo e para usá-las com contadores. O laço entre 2060 e 2100 desenha as linhas de longitude, e o que fica entre 2150 e 2190 traça as linhas de latitude.

O eixo do globo corta os pólos e sua direção é determinada pelas variáveis cujos valores definimos no começo do programa. Para verificar como os pólos do globo são posicionados em relação aos eixos da tela, estabeleça zero para duas das coordenadas da posição do observador (0, 0, 200, por exemplo). Depois, defina valores positivos ou negativos para as três coordenadas e veja como os pólos se movimentam em relação aos eixos.

DO GLOBO AO TORO

O globo é uma das formas mais simples; para desenhá-lo, recorremos a uma série de elipses, mas o mesmo programa contém todos os elementos necessários para o desenho de uma forma mais exótica — um toro (ou argola). Para realizá-lo, precisamos apenas de pequenas modificações na rotina do globo, como mostra a listagem a seguir:

```
150 R = 10:N = 18:RT = 20:N2 =
18: GOSUB 2000
2040 KA = 2 * PI / N2:NC = N2
2045  IF RT = 0 THEN NC =  INT
(NC / 2)
2060  FOR KC = 1 TO NC
2080 XS = RT:YS = 0: GOSUB 6000

2130 KA = 2 * PI / N
2135  IF RT = 0 THEN KA = KA /
2
2170 XS = 0:YS = 0:R = RT + RR
*  SIN (KB):N = N2: GOSUB 6000
```

```
150 R=10:N=18:RT=20:N2=18:GOSUB
 2000
2040 KA=2*PI/N2:NC=N2
2045 IF RT=0 THEN NC=INT(NC/2)
2060 FOR KC=1 TO NC
2080 XS=RT:YS=0:GOSUB 6000
2130 KA=2*PI/N
2135 IF RT=0 THEN KA=KA/2
2170 XS=0:YS=0:R=RT+RR*SIN(KB):
N=N2:GOSUB 6000
```

```
 150 LET R=10: LET N=18: LET RT
 =20: LET N2=18: GOSUB 2000
2040 LET KA=2*PI/N2: LET NC=N2
2045 IF RT=0 THEN  LET NC=INT (
NC/2)
2060 FOR K=1 TO NC
2080 LET XS=RT: LET YS=0: GOSUB
 6000
2130 LET KA=2*PI/N
2135 IF RT=0 THEN  LET KA=KA/2
2170 LET XS=0: LET YS=0: LET R=
RT+RR*SIN KB: LET N=N2: GOSUB
 6000
```

Rode o programa e defina valores para **D** e para a posição do observador (1000, 200, 200, 200, por exemplo). Desta vez, **R** contém o raio interno do tubo, e o número de anéis é determinado por **N**. Esses anéis são equivalentes às linhas de longitude no globo. **RT** especifica a distância entre a origem e o tubo, e **N2** determina o número de linhas que formam o tubo, o que equivale às

P&R

Há alguma forma de se acelerar o desenho da estação espacial?

O modo mais grosseiro de se acelerar o programa consiste em reduzir o número de etapas nas rotinas principais — a rotina do círculo na estação espacial e a da grade nos cubos. Contudo, esse método reduz a suavidade das elipses e altera a estrutura do desenho. Uma alternativa melhor seria diminuir o número de acessos a essas rotinas.

Se quisermos, por exemplo, diminuir o número de segmentos que cortam o tubo de um toro, devemos reduzir os valores de **R**, **N**, **RT** e **N2** na rotina do toro que começa na linha 2000. É necessário também que as rotinas mais usadas estejam no começo do programa e que se usem variáveis em vez de números, sempre que possível. O uso indiscriminado de comandos **GOTO** também desacelera a execução de um programa.

Talvez a maneira mais fácil de acelerar esse programa seja omitir o desenho dos tubos que ligam o toro ao globo; faremos isso com um simples desvio (**GOTO**) no programa.

linhas de latitude no globo. O valor de **RT** deve ser maior que o de **R**, mas a rotina funcionará bem mesmo se isso não acontecer. Altere o valor dessas variáveis e observe o efeito obtido. Se **RT** for zero e **N** igual a **N2**, o toro se transformará numa esfera, como aquela da rotina do globo.

COMO COMBINAR IMAGENS

Não seria difícil substituir a rotina do cubo que desenha quatro imagens juntas na tela pela do globo ou do toro. Do mesmo modo, também seria simples chamar qualquer combinação das três rotinas e desenhar, por exemplo: cubos dentro de um globo ou de uma argola ou mesmo um sistema em que uma argola contenha um globo que, por sua vez, contenha um cubo. Poderíamos ainda usar declarações **GOTO** em certas linhas para suprimir os dois últimos lados, tornando assim menos confuso o desenho. Digite essas linhas e rode o programa para ver surgir uma imponente estação espacial.

```
150  GOSUB 3000
3000 R = 6:N = 12:RT = 40:N2 =
24: GOSUB 2000
3030  FOR F = 1 TO 6
3040 A = (F + .25) * PI / 3:L1
= 10:L2 = 34:N1 = 12:R = 2:N2 =
 8: GOSUB 3500
3050  NEXT
3060 R = 10:N = 12:RT = 0:N2 =
12: GOSUB 2000
3070  RETURN
3500 SA =  SIN (A)
3530 CA =  COS (A)
3540 T1 =  - SA:T2 = CA:T3 = 0
3550 T4 = 0:T5 = 0:T6 = 1
3560 EA = (L2 - L1) / N1
3570 EB = L1
3580  FOR EC = 0 TO N1
3590 T7 = EB * CA:T8 = EB * SA:
T9 = 0
3600 XS = 0:YS = 0:N = N2: GOSU
B 6000
3610 EB = EB + EA
3620  NEXT
3630 T1 = CA:T2 = SA:T3 = 0
3640  FOR EC = 1 TO N2
3650 EA = EC * PI / 8
3660 T7 = R * SA *  COS (EA):T8
 =  - R * CA *  COS (EA):T9 = R
 *  SIN (EA)
3670 XS = L1:YS = 0:XE = L2:YE
= 0: GOSUB 9500
3680  NEXT
3690  RETURN
```

```
150 GOSUB 3000
3000 R=6:N=12:RT=40:N2=24:GOSUB
 2000
3030 FOR F=1 TO 6
3040 A=(F+.25)*PI/3:L1=10:L2=34
:N1=12:R=2:N2=8:GOSUB 3500
3050 NEXT
3060 R=10:N=12:RT=0:N2=12:GOSUB
 2000
3070 RETURN
3500 SA=SIN(A)
3530 CA=COS(A)
3540 T1=-SA:T2=CA:T3=0
3550 T4=0:T5=0:T6=1
3560 EA=(L2-L1)/N1
3570 EB=L1
3580 FOR EC=0 TO N1
3590 T7=EB*CA:T8=EB*SA:T9=0
3600 XS=0:YS=0:N=N2:GOSUB 6000
3610 EB=EB+EA
3620 NEXT
3630 T1=CA:T2=SA:T3=0
3640 FOR EC=1 TO N2
3650 EA=EC*PI/8
3660 T7=R*SA*COS(EA):T8=-R*CA*C
OS(EA):T9=R*SIN(EA)
3670 XS=L1:YS=0:XE=L2:YE=0:GOSU
B 9500
3680 NEXT
3690 RETURN
```

```
150 GOSUB 3000
3000 R=6:N=12:RT=40:N2=24:GOSUB
 2000
3030 FOR F=1 TO 6
3040 A=(F+.25)*PI/3:L1=10:L2=34
:N1=12:R=2:N2=8:GOSUB 3500
3050 NEXT
3060 R=10:N=12:RT=0:N2=12:GOSUB
 2000
3070 RETURN
3500 SA=SIN(A)
3530 CA=COS(A)
3540 T1=-SA:T2=CA:T3=0
3550 T4=0:T5=0:T6=1
3560 EA=(L2-L1)/N1
3570 EB=L1
3580 FOR EC=0 TO N1
3590 T7=EB*CA:T8=EB*SA:T9=0
3600 XS=0:YS=0:N=N2:GOSUB 6000
3610 EB=EB+EA
3620 NEXT
3630 T1=CA:T2=SA:T3=0
3640 FOR EC=1 TO N2
3650 EA=EC*PI/8
3660 T7=R*SA*COS(EA):T8=-R*CA*C
OS(EA):T9=R*SIN(EA)
3670 XS=L1:YS=0:XE=L2:YE=0:GOSU
B 9500
3680 NEXT
3690 RETURN
```

```
 150 GOSUB 3000
3000 LET R=6: LET N=24: LET RT=
40: LET N2=24: GOSUB 2000
3030 FOR F=1 TO 6
3040 LET A=(F+.25)*PI/3: LET L1
=10: LET L2=34: LET N1=12: LET
R=2: LET N2=8: GOSUB 3500
3050 NEXT F
3060 LET R=10: LET N=12: LET RT
=0: LET N2=12: GOSUB 2000
3070 RETURN
3500 LET SA=SIN A
3530 LET CA=COS A
3540 LET T1=-SA: LET T2=CA: LET
 T3=0
3550 LET T4=0: LET T5=0: LET T6
=1
3560 LET EA=(L2-L1)/N1
3570 LET EB=L1
3580 FOR E=0 TO N1
3590 LET T7=EB*CA: LET T8=EB*SA
: LET T9=0
3600 LET XS=0: LET YS=0: LET N=
N2: GOSUB 6000
3610 LET EB=EB+EA
3620 NEXT E
3630 LET T1=CA: LET T2=SA: LET
T3=0
3640 FOR E=1 TO N2
3650 LET EA=E*PI/8
3660 LET T7=R*SA*COS EA: LET T8
=-R*CA*COS EA: LET T9=R*SIN EA
3670 LET XS=L1: LET YS=0: LET X
E=L2: LET YE=0: GOSUB 9500
3680 NEXT E
3690 RETURN
```

MICRO DICAS

COMO MELHORAR A NITIDEZ

Quando a imagem dos quatro cubos é pequena — como acontece sempre que os valores para a posição do observador (X, Y e Z) são bem maiores que o da distância de observação **D** — torna-se difícil perceber o que se passa durante a execução do desenho. Pode-se melhorar a nitidez inserindo declarações **GOTO** em locais apropriados ao longo do programa (entre as linhas 1000 e 1170). A mais simples dessas declarações seria **GOTO 1220**, inserida na linha 1125 (linha nova) do programa dos quatro cubos.

A extensão de experimentos possíveis com o programa de toro é ilimitada. Tente alguns números pequenos para **D** (de 100 a 500, digamos), juntamente com diversos valores para X, Y e Z; você obterá, assim, diferentes perspectivas (especialmente porque os segmentos do toro que estiverem muito próximos serão omitidos). Tente também valores grandes para **D** e observe o que acontece com a distorção.

Agora, como simples desafio, poderíamos dar um toque final ao programa, adicionando cor ao fundo, por exemplo. Deixaremos por conta de sua criatividade incluir estrelas e planetas ao redor da estação.

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR LINHA

LINHA	FABRICANTE	MODELO
Apple II +	Appletronica	Thor 2010
Apple II +	CCE	MC-4000 Exato
Apple II +	CPA	Absolutus
Apple II +	CPA	Polaris
Apple II +	Digitus	DGT-AP
Apple II +	Dismac	D-8100
Apple II +	ENIAC	ENIAC II
Apple II +	Franklin	Franklin
Apple II +	Houston	Houston AP
Apple II +	Magnex	DM II
Apple II +	Maxitronica	MX-2001
Apple II +	Maxitronica	MX-48
Apple II +	Maxitronica	MX-64
Apple II +	Maxitronica	Maxitronic I
Apple II +	Microcraft	Craf II Plus
Apple II +	Milmar	Apple II Plus
Apple II +	Milmar	Apple Master
Apple II +	Milmar	Apple Senior
Apple II +	Omega	MC-400
Apple II +	Polymax	Maxxi
Apple II +	Polymax	Poly Plus
Apple II +	Spectrum	Microengenho I
Apple II +	Spectrum	Spectrum ed
Apple II +	Suporte	Venus II
Apple II +	Sycomig	SIC I
Apple II +	Unitron	AP II
Apple II +	Victor do Brasil	Elppa II Plus
Apple II +	Victor do Brasil	Elppa Jr.
Apple IIe	Microcraft	Craft IIe
Apple IIe	Microdigital	TK-3000 IIe
Apple IIe	Spectrum	Microengenho II
MSX	Gradiente	Expert GPC-1
MSX	Sharp	Hotbit HB-8000
Sinclair Spectrum	Microdigital	TK-90X
Sinclair Spectrum	Timex	Timex 2000
Sinclair ZX-81	Apply	Apply 300
Sinclair ZX-81	Engebras	AS-1000
Sinclair ZX-81	Filcres	NEZ-8000
Sinclair ZX-81	Microdigital	TK-82C
Sinclair ZX-81	Microdigital	TK-83
Sinclair ZX-81	Microdigital	TK-85
Sinclair ZX-81	Prologica	CP-200
Sinclair ZX-81	Ritas	Ringo R-470
Sinclair ZX-81	Timex	Timex 1000
Sinclair ZX-81	Timex	Timex 1500
TRS-80 Mod. I	Dismac	D-8000
TRS-80 Mod. I	Dismac	D-8001/2
TRS-80 Mod. I	LNW	LNW-80
TRS-80 Mod. I	Video Genie	Video Genie I
TRS-80 Mod.III	Digitus	DGT-100
TRS-80 Mod.III	Digitus	DGT-1000
TRS-80 Mod.III	Kemitron	Naja 800
TRS-80 Mod.III	Prologica	CP-300
TRS-80 Mod.III	Prologica	CP-500
TRS-80 Mod.III	Sysdata	Sysdata III
TRS-80 Mod.III	Sysdata	Sysdata Jr.
TRS-80 Mod.IV	Multix	MX-Compacto
TRS-80 Mod.IV	Sysdata	Sysdata IV
TRS-Color	Codimex	CS-6508
TRS-Color	Dynacom	MX-1600
TRS-Color	LZ	Color 64
TRS-Color	Microdigital	TKS-800
TRS-Color	Prologica	CP-400

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR NOME

FABRICANTE	MODELO	PAÍS	LINHA
Appletronica	Thor 2010	Brasil	Apple II +
Apply	Apply 300	Brasil	Sinclair ZX-81
CCE	MC-4000 Exato	Brasil	Apple II +
CPA	Absolutus	Brasil	Apple II +
CPA	Polaris	Brasil	Apple II +
Codimex	CS-6508	Brasil	TRS-Color
Digitus	DGT-100	Brasil	TRS-80 Mod.III
Digitus	DGT-1000	Brasil	TRS-80 Mod.III
Digitus	DGT-AP	Brasil	Apple II +
Dismac	D-8000	Brasil	TRS-80 Mod. I
Dismac	D-8001/2	Brasil	TRS-80 Mod. I
Dismac	D-8100	Brasil	Apple II +
Dynacom	MX-1600	Brasil	TRS-Color
ENIAC	ENIAC II	Brasil	Apple II +
Engebras	AS-1000	Brasil	Sinclair ZX-81
Filcres	NEZ-8000	Brasil	Sinclair ZX-81
Franklin	Franklin	USA	Apple II +
Gradiente	Expert GPC1	Brasil	MSX
Houston	Houston AP	Brasil	Apple II +
Kemitron	Naja 800	Brasil	TRS-80 Mod.III
LNW	LNW-80	USA	TRS-80 Mod. I
LZ	Color 64	Brasil	TRS-Color
Magnex	DM II	Brasil	Apple II +
Maxitronica	MX-2001	Brasil	Apple II +
Maxitronica	MX-48	Brasil	Apple II +
Maxitronica	MX-64	Brasil	Apple II +
Maxitronica	Maxitronic I	Brasil	Apple II +
Microcraft	Craft II Plus	Brasil	Apple II +
Microcraft	Caft IIe	Brasil	Apple IIe
Microdigital	TK-3000 IIe	Brasil	Apple IIe
Microdigital	TK-82C	Brasil	Sinclair ZX-81
Microdigital	TK-83	Brasil	Sinclair ZX-81
Microdigital	TK-85	Brasil	Sinclair ZX-81
Microdigital	TK-90X	Brasil	Sinclair Spectrum
Microdigital	TKS-800	Brasil	TRS-Color
Milmar	Apple II Plus	Brasil	Apple II +
Milmar	Apple Master	Brasil	Apple II +
Milmar	Apple Senior	Brasil	Apple II +
Multix	MX-Compacto	Brasil	TRS-80 Mod.IV
Omega	MC-400	Brasil	Apple II +
Polymax	Maxxi	Brasil	Apple II +
Polymax	Poly Plus	Brasil	Apple II +
Prologica	CP-200	Brasil	Sinclair ZX-81
Prologica	CP-300	Brasil	TRS-80 Mod.III
Prologica	CP-400	Brasil	TRS-Color
Prologica	CP-500	Brasil	TRS-80 Mod.III
Ritas	Ringo R-470	Brasil	Sinclair ZX-81
Sharp	Hotbit HB-8000	Brasil	MSX
Spectrum	Microengenho I	Brasil	Apple II +
Spectrum	Microengenho II	Brasil	Apple IIe
Spectrum	Spectrum ed	Brasil	Apple II +
Suporte	Venus II	Brasil	Apple II +
Sycomig	SIC I	Brasil	Apple II +
Sysdata	Sysdata III	Brasil	TRS-80 Mod.III
Sysdata	Sysdata IV	Brasil	TRS-80 Mod.IV
Sysdata	Sysdata Jr.	Brasil	TRS-80 Mod.III
Timex	Timex 1000	USA	Sinclair ZX-81
Timex	Timex 1500	USA	Sinclair ZX-81
Timex	Timex 2000	USA	Sinclair Spectrum
Unitron	AP II	Brasil	Apple II +
Victor do Brasil	Elppa II Plus	Brasil	Apple II +
Victor do Brasil	Elppa Jr.	Brasil	Apple II +
Video Genie	Video Genie I	USA	TRS-80 Mod. I

UM LOGOTIPO PARA CADA MODELO DE COMPUTADOR

INPUT foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:

Sinclair ZX-81

TRS-80

TK-2000

MSX

Spectrum

TRS-Color

Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

NO PRÓXIMO NÚMERO

APLICAÇÕES

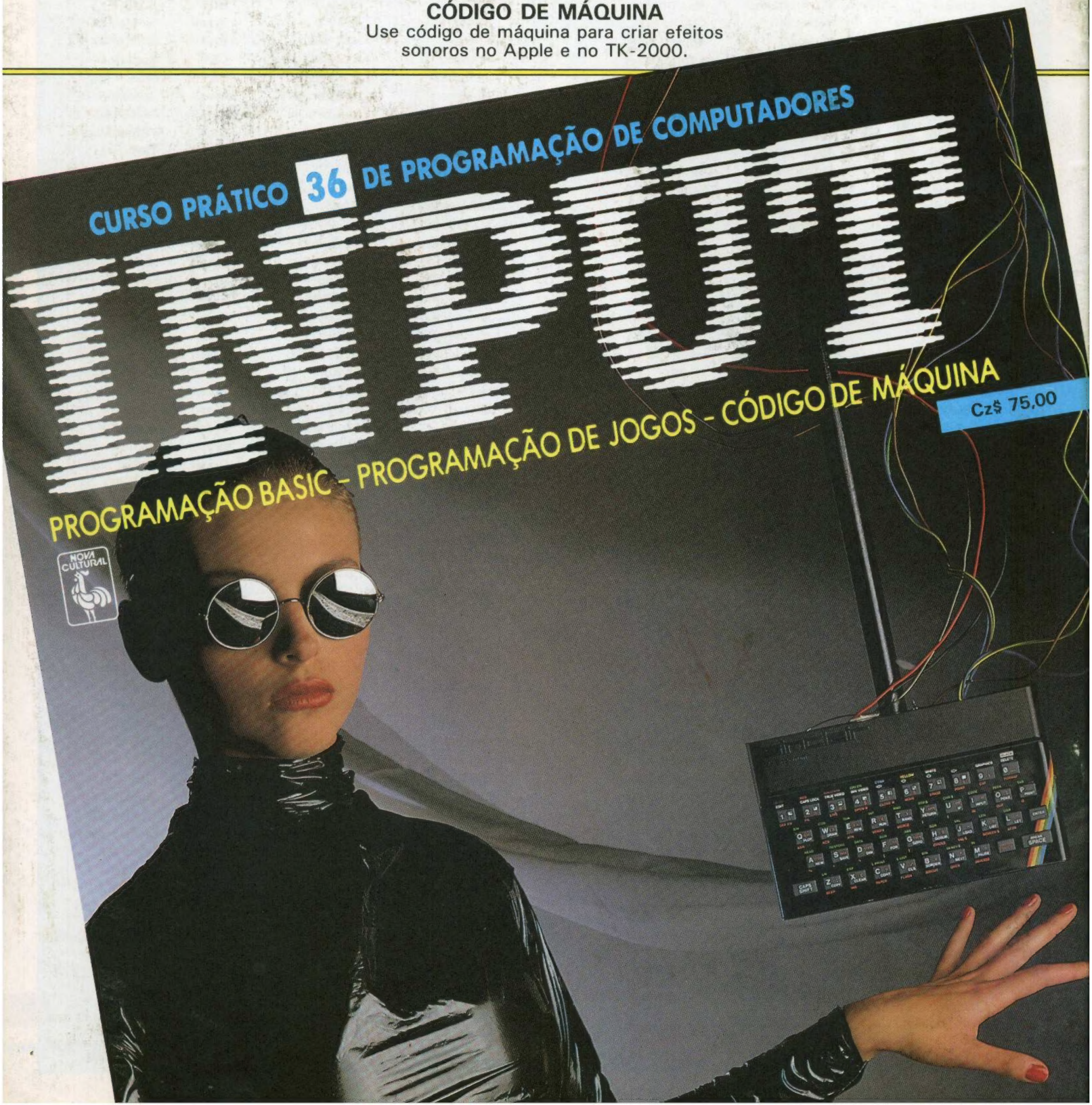
Novas rotinas para o seu banco de dados que lhe permitirão organizar melhor as informações.

PROGRAMAÇÃO DE JOGOS

Um jogo de adivinhação de palavras inspirado no ''jogo da forca''. Muito versátil, ele agrada a todas as idades.

PROGRAMAÇÃO BASIC

Comandos que permitem ao micro olhar para a sua própria tela: úteis, sobretudo, para jogos em que ocorrem colisões.

CÓDIGO DE MÁQUINA

Use código de máquina para criar efeitos sonoros no Apple e no TK-2000.